Towards Rigour:
A Response and Proposal for Methodological Extension

Duncan J. McGreggor

Working Paper — Draft for Discussion

February 2026

Abstract

This paper responds to the critical assessment of our LLM-mediated knowledge consolidation pipeline (McGreggor, 2026). We accept the commentary's central claims: that the pipeline independently rediscovers established principles from ontology engineering, library science, and cognitive architecture; that several significant methodological practices are absent; and that the novel contributions identified—particularly the dual-format concept card and procedural guide synthesis—warrant formalisation. We address each identified omission in turn, distinguishing between those we accept as immediate priorities (competency questions, confidence scoring, expanded relation inventory), those we accept with methodological caveats (OntoClean, authority control), and those we partially contest (ontology reuse in the specific context of LLM-mediated extraction). We then propose a concrete set of methodological extensions—a "v2" pipeline—incorporating the commentary's recommendations alongside insights from our subsequent reading of the literature it cites. We present an implementation roadmap organised by priority and dependency, with the aim of transforming the pipeline from a working system into a validated methodology.

1. Introduction

We are grateful for the critical assessment presented in the companion commentary. The assessment accomplishes precisely what we hoped to provoke by publishing the pipeline paper in its current, admittedly under-theorised form: it situates our practical work within the relevant intellectual traditions, identifies the theoretical commitments our design decisions encode, and provides a clear map of what we have missed.

The commentary's most valuable contribution is not any single critique but the overall frame it provides. We built the pipeline to solve a practical problem—getting well-grounded responses from an LLM—and we were aware that we had not situated the solution theoretically. What we were not fully aware of was *how many* distinct research traditions our pipeline touches, or how precisely some of our pragmatic decisions map

onto established principles. Learning that our concept-card-to-guide architecture mirrors ACT-R's declarative-procedural distinction, or that our prerequisite graph is one axiom short of a formal knowledge space, transforms our understanding of what we have built and sharpens our sense of what it needs.

This response is organised in three parts. First, we address the identified omissions, accepting most and proposing implementation approaches for each. Second, we propose a revised pipeline architecture—"v2"—that incorporates these extensions while preserving the pragmatic strengths of the current system. Third, we present an implementation roadmap that sequences these changes by priority and dependency.

2. Addressing the Identified Omissions

2.1 Competency Questions: Accepted as Highest Priority

The commentary identifies the absence of competency questions as the pipeline's most consequential omission. We agree unreservedly. In retrospect, the omission is particularly striking because we *implicitly* employed competency questions during development—every test of the system involved posing questions to the LLM and evaluating whether the knowledge base supported adequate answers. What we failed to do was formalise this process: to write down the questions, to organise them systematically, and to use them as both specification and acceptance criteria.

We propose to address this by introducing a **Phase 0** before any extraction begins. For each domain to which the pipeline is applied, we will elicit and document 30–50 competency questions spanning the range of query types the knowledge base should support. Following Wisniewski et al.'s (2019) analysis, we will classify these questions by type: definitional ("What is X?"), relational ("How does X relate to Y?"), procedural ("How do I accomplish Z?"), prerequisite ("What must I know before X?"), and diagnostic ("What distinguishes X from Y?"). After extraction, each competency question will be tested against the knowledge base, and failures will drive targeted re-extraction or guide creation.

Critically, we recognise that competency questions are not a one-time exercise. As the knowledge base grows and as new use cases emerge, the question set must be extended. We will maintain the CQ catalogue as a living document, versioned alongside the knowledge base, with each pipeline run reporting coverage metrics against the current question set.

2.2 Confidence Scoring: Accepted with Implementation Proposal

The absence of confidence scoring is, on reflection, a significant gap that has been masked by the high quality of the primary sources we have processed. Music theory textbooks are authoritative, well-edited, and internally consistent. Extraction from less curated sources—conference proceedings, online tutorials, historical texts—would expose the need for confidence signals immediately.

We propose a two-tier approach. First, at extraction time, we will implement **multi-generation consistency checking**—extracting each concept card two to three times with varied prompts and comparing the results. Consistent extractions receive high confidence; divergent extractions are flagged for human review. This approach is inspired by SelfCheckGPT but adapted for structured extraction rather than free-text generation.

Second, at the knowledge graph level, we will implement **structural confidence propagation**: concepts with multiple independent source attestations receive higher structural confidence than those attested by a single source. Relationships confirmed by multiple concept cards receive higher confidence than those asserted by a single card's "Related Concepts" section. This mirrors the approach used by NELL (Never-Ending Language Learner), which assigns confidence scores based on the number and diversity of extraction patterns that support each assertion.

2.3 Expanded Relation Inventory: Accepted with Staged Implementation

The commentary's observation that three edge types are insufficient is correct, and we have encountered the limitation in practice. The "related" edge type has indeed become a catch-all, and queries that require distinguishing part-whole relationships from contrast relationships from scaffolding relationships currently cannot be expressed.

We propose expanding to ten typed relations in two stages. Stage one introduces the relations for which we have immediate, demonstrated need: **part_of / has_part** (for structural decomposition), **instance_of / exemplifies** (for connecting abstract concepts to concrete examples), and **contrasts_with** (for disambiguation of commonly confused concepts). Stage two, contingent on successful application in a second domain, adds: **scaffolds** (weaker than prerequisite), **same_as** (for cross-source entity resolution), and **temporal_sequence** (for procedural ordering within guides).

Each relation will be defined with explicit domain and range constraints, cardinality expectations, and semantic axioms. For example: prerequisite_of must be acyclic (enforceable by topological sort); part_of must be irreflexive and antisymmetric;

contrasts_with must be symmetric. These constraints enable the structural validation that the commentary rightly identifies as absent.

2.4 Ontology Evaluation: Accepted with Methodological Caveats

We accept the need for systematic evaluation and recognise OntoClean, OOPS!, and Paulheim's refinement framework as the relevant tools. However, we wish to note a tension between these formal evaluation approaches and the pipeline's design philosophy.

OntoClean requires assigning metaproperties—rigidity, identity, unity—to each concept category. This is a deeply philosophical exercise for a domain like music theory, where the ontological status of concepts is contested. Is "sonata form" a rigid concept (necessarily what it is in all possible worlds) or a non-rigid one (a convention that could have been otherwise)? We do not object to the exercise, but we note that the answers may be domain-dependent, debatable, and subject to revision. We will implement OntoClean analysis as a *diagnostic tool* rather than a *gate*: it will flag potential taxonomic inconsistencies for review rather than blocking pipeline execution.

OOPS! pitfall scanning, by contrast, is more straightforwardly applicable. Many of its 41 pitfall patterns correspond to structural errors that are unambiguously wrong (cycles in strict hierarchies, disconnected concept islands, missing domain constraints) and can be checked automatically. We will integrate OOPS!-style structural validation as an automated step after graph construction, with results reported as part of the pipeline's quality metrics.

2.5 Authority Control: Accepted as Essential for Retrieval

The commentary's point about syndetic structure and authority control is practically urgent. We have observed retrieval failures in the existing system when users employ variant terminology. A query for "V7 chord" should surface the concept card for "dominant seventh chord"; currently, it may not. This is a solved problem in library science, and we should not re-solve it from first principles.

We propose implementing authority control through a **variant registry**—a supplementary data structure that maps each canonical concept identifier to its known variant forms (including abbreviations, informal names, symbolic notation, and common misspellings). This registry will be populated during extraction (LLMs are surprisingly effective at generating variant forms when prompted) and refined through usage analysis (logging queries that fail to match any concept and manually mapping them to the intended target).

The full-text search index will be extended to include variant forms as searchable aliases, ensuring that any known variant surfaces the canonical concept. This is

functionally equivalent to implementing USE/UF cross-references in a controlled vocabulary.

2.6 Knowledge Space Axiom Verification: Accepted as Theoretically Important

We accept the recommendation to verify closure under union and acknowledge that we had not been aware of this specific requirement before encountering Doignon and Falmagne's work through the commentary's references. The practical implication is clear: if two independently valid learning paths, when combined, produce a knowledge state whose prerequisites are not mutually satisfied, the prerequisite graph contains an inconsistency.

Implementing full KST verification is computationally non-trivial for large concept sets (the space of feasible states grows exponentially). We propose a *sampling-based approach*: for each concept with multiple prerequisite paths, we will verify that the union of the paths' endpoints constitutes a feasible state. This does not provide exhaustive verification but will catch the most common violations—those at path convergence points.

2.7 Ontology Reuse: Partially Contested

This is the one recommendation we accept with significant reservation. The commentary is correct that existing ontological resources exist for music theory and that NeOn Methodology identifies reuse as a core scenario. However, our experience suggests that the *alignment cost* for existing music ontologies is substantial relative to the benefit.

The Music Ontology (Raimond et al.) is oriented toward music information retrieval—describing recordings, performances, and audio features—rather than music theory pedagogy. The conceptual overlap with our knowledge base is real but partial: both represent "chord" as a concept, but our representation focuses on theoretical properties, construction rules, and pedagogical context, while the Music Ontology focuses on acoustic properties and performance metadata. Aligning the two would require extensive mapping work with uncertain return.

We will pursue a **selective alignment** approach: where our concepts have clear, unambiguous counterparts in existing ontologies, we will record the mapping (using *skos:exactMatch* or *skos:closeMatch*). Where the mapping is contested or partial, we will document the relationship without asserting formal equivalence. This preserves interoperability benefits without forcing conceptual commitments that our specific use case does not require.

3. Proposed Pipeline Architecture: Version 2

Incorporating the accepted recommendations, we propose the following revised pipeline. Figure 1 presents the complete v2 architecture, with new elements highlighted. The fundamental structure—source acquisition, concept extraction, guide synthesis, graph construction, and unified search—is preserved. The additions are: a requirements phase (Phase 0), validation and evaluation mechanisms, enriched metadata, and feedback loops between later and earlier stages.

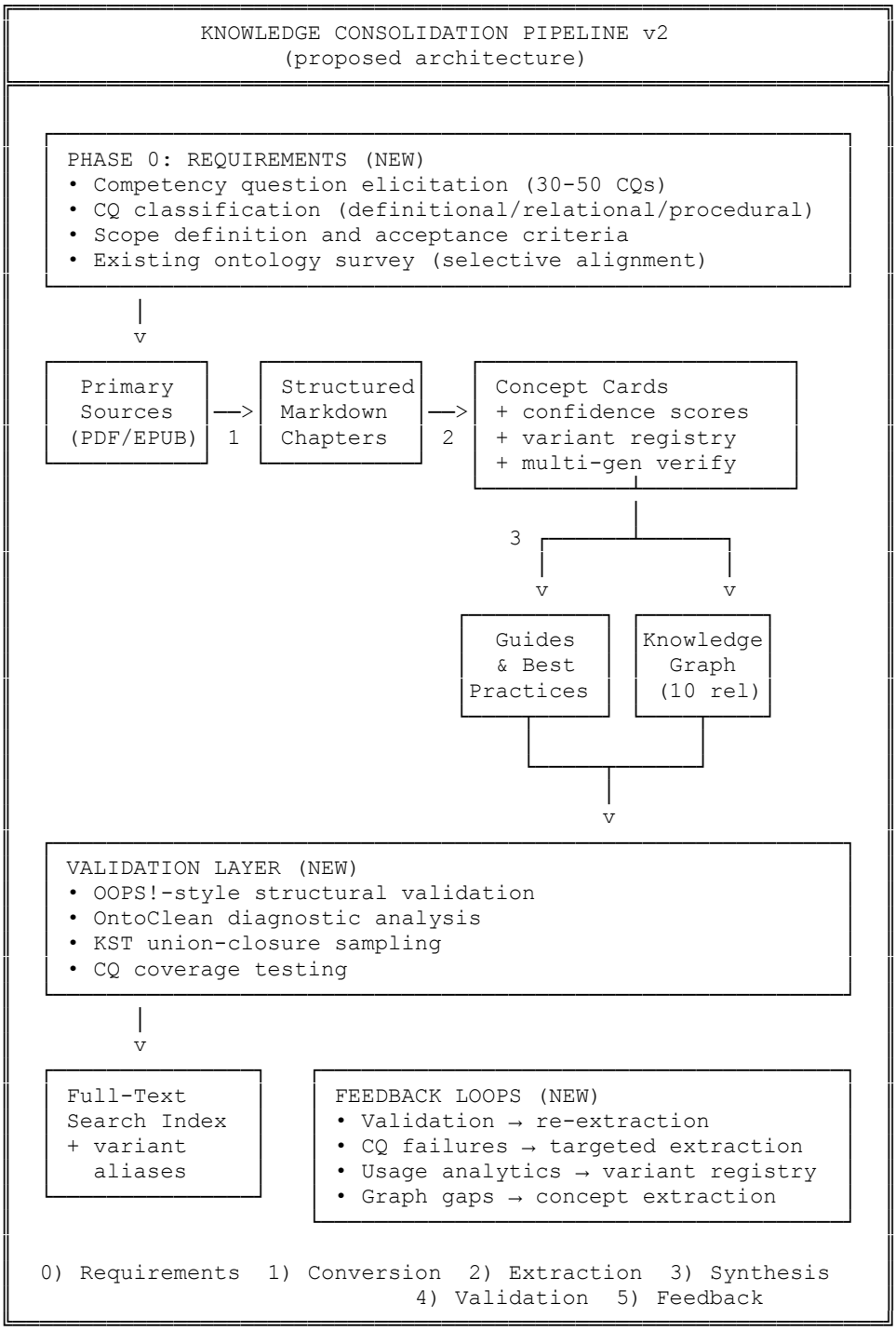

Figure 1. Proposed v2 pipeline architecture. New elements (Phase 0, validation layer, enriched feedback loops, metadata) are marked. The core transformation stages are preserved.

3.1 Phase 0: Requirements and Scope

The most significant structural change is the addition of a requirements phase before any extraction. This phase produces three artifacts: a competency question catalogue (classified by type), a scope definition (which areas of the domain are in and out of scope), and the results of an existing ontology survey (identifying candidates for selective alignment). These artifacts collectively define what the knowledge base must accomplish and provide the acceptance criteria against which it will be evaluated.

We note that this phase can itself be LLM-mediated. Given a set of source materials and a high-level domain description, an LLM can generate candidate competency questions that a domain expert then reviews, refines, and prioritises. This preserves the pragmatic character of the pipeline while introducing the formal requirements that the commentary correctly identifies as missing.

3.2 Enriched Concept Cards

The concept card structure is extended in three ways. First, each card includes a **confidence score** derived from multi-generation consistency checking during extraction. Second, each card includes a **variant registry**—a list of known alternative names, abbreviations, and notational forms for the concept. Third, the YAML frontmatter is formalised using **LinkML** schema definitions (the same framework used by OntoGPT/SPIRES), enabling automatic validation and optional OWL export.

The variant registry addresses the authority control gap directly. During extraction, the LLM is prompted to identify all variant forms of each concept name. These variants are indexed in the full-text search system as aliases, ensuring retrieval regardless of the user's preferred terminology.

3.3 Expanded and Constrained Knowledge Graph

The knowledge graph is extended from three to ten relation types, each with explicit semantic constraints. Figure 2 presents the expanded relation inventory with constraints.

RELATION INVENTORY v2

RELATION	CONSTRAINTS	STAGE
prerequisite_of	acyclic, directed	Stage 1 (existing)
extends	acyclic, directed	Stage 1 (existing)
related	symmetric	Stage 1 (existing)
part_of/has_part	irreflexive, anti-symmetric, transitive	Stage 1 (new)
instance_of	directed	Stage 1 (new)
contrasts_with	symmetric	Stage 1 (new)
scaffolds	acyclic, directed	Stage 2 (new)
same_as	symmetric, transitive	Stage 2 (new)
temporal_sequence	acyclic, directed	Stage 2 (new)

Figure 2. Expanded relation inventory with semantic constraints and implementation staging.

3.4 Validation Layer

The most architecturally significant addition is a dedicated validation layer that operates after graph construction and before search indexing. This layer performs four categories of validation: structural checks (OOPS!-style pitfall detection), taxonomic analysis (OntoClean diagnostic), mathematical verification (KST union-closure sampling), and functional testing (CQ coverage evaluation). Results are reported as quality metrics, with critical failures blocking downstream indexing and non-critical issues logged for human review.

3.5 Feedback Loops

The v1 pipeline is essentially unidirectional: information flows from source to search index with no formalised mechanism for later stages to inform earlier ones. The commentary's concern about this is well-founded, and the current system's informal practice of re-extracting concepts when graph construction reveals gaps confirms the need.

The v2 architecture formalises four feedback loops. First, validation failures trigger **targeted re-extraction**: if a structural check reveals a missing prerequisite or an orphaned concept, the extraction stage is re-run for the relevant source material with explicit prompting for the missing elements. Second, CQ failures drive **gap-targeted extraction**: if a competency question cannot be answered from the knowledge base, the source material is re-analysed specifically for the concepts needed to answer it. Third, search analytics feed the **variant registry**: queries that fail to match any concept are logged, reviewed, and mapped to canonical concepts as new variants. Fourth, graph structure analysis identifies **extraction candidates**: disconnected components, low-centrality concepts, and missing bridge concepts suggest areas where additional extraction or source material may be needed.

4. Implementation Roadmap

We propose a three-phase implementation schedule, organised by dependency and priority. Figure 3 presents the phased roadmap.

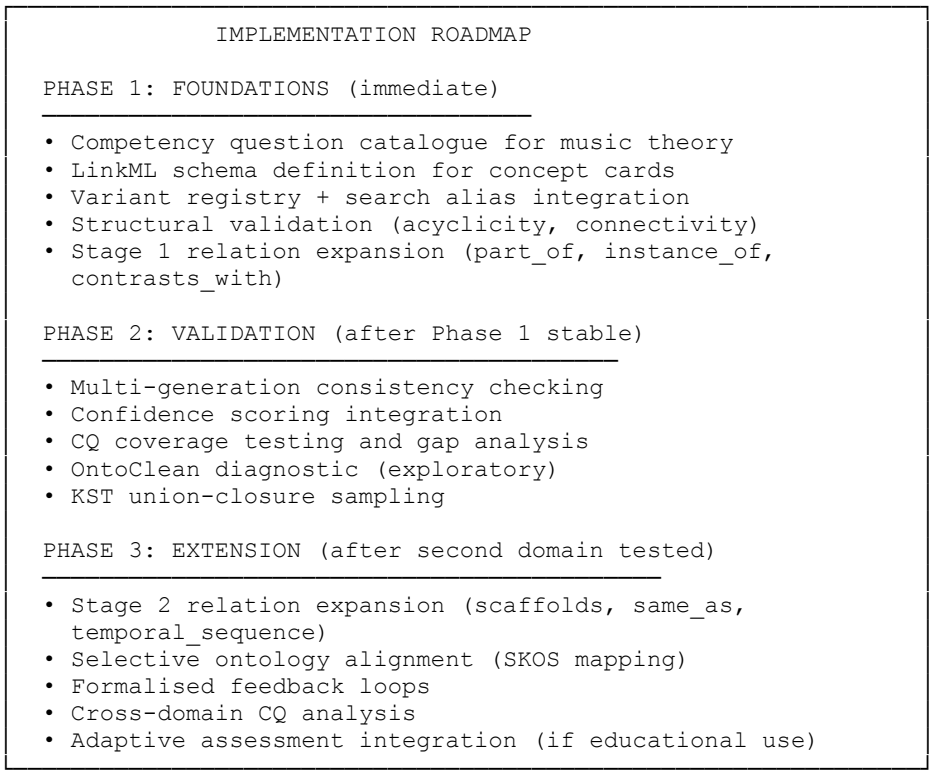

Figure 3. Three-phase implementation roadmap, organised by dependency and priority.

Phase 1 addresses the most immediately impactful gaps: competency questions (which define what the system should do), schema formalisation (which enables automated validation), variant registries (which fix known retrieval failures), and the first tranche of expanded relations (which address the most pressing expressivity limitations). These changes can be implemented against the existing music theory knowledge base and evaluated immediately.

Phase 2 introduces the validation and quality assurance mechanisms that transform the pipeline from a working system into a validated one. These depend on Phase 1's schema formalisation (which provides the structural definitions that validation checks against) and competency questions (which provide the functional tests). Multi-generation consistency checking is the highest-priority item in this phase, as it provides the confidence signals needed for responsible extension to less curated source material.

Phase 3 is contingent on successful application of the v2 pipeline to a second domain. The remaining relation types, ontology alignment, and cross-domain analysis only become meaningful when the pipeline has been demonstrated to generalise beyond its initial application. We resist the temptation to design for generality before we have evidence of it.

5. On What the Pipeline Contributes

The commentary identifies three elements of the pipeline as potentially novel: the dual-format concept card, procedural guide synthesis from declarative knowledge bases, and MCP as a knowledge service interface. We wish to comment briefly on each.

The dual-format card was, as the commentary surmises, a pragmatic invention. We needed the LLM to produce both structured metadata and readable exposition, and early attempts to separate these into distinct extraction passes produced lower-quality results than a single, integrated extraction. The finding by Tam et al. (2024) that strict JSON constraints degrade reasoning performance provides a theoretical explanation for what we observed empirically: the Markdown body provides a "reasoning space" that improves the quality of the structured YAML output. We plan to investigate this relationship more formally by comparing extraction quality across single-pass (integrated card) and two-pass (separate structure and prose) approaches.

Guide synthesis is the element we are most interested in formalising. The current process is largely manual—we identify a topic that warrants a guide, select the relevant concept cards, and prompt the LLM to compose them into an integrated document. Automating the identification of guide-worthy topic clusters (perhaps via community detection in the knowledge graph) and the composition process itself would be a meaningful contribution to the procedural knowledge synthesis literature.

MCP as a knowledge service interface is indeed emergent, and we are conscious of operating at the leading edge of a rapidly evolving standard. Our implementation provides a proof of concept for a pattern that we believe will become standard: structured knowledge bases served through tool-level protocols that LLMs can invoke during interactive sessions. The formalisation of this pattern—defining what operations a knowledge service should expose, what contracts it should satisfy, and what guarantees it should provide—is work we intend to contribute to as the MCP ecosystem matures.

6. Conclusion

The critical assessment has accomplished its intended purpose: it has situated our pragmatic work within the relevant theoretical landscape, identified the specific practices we should adopt, and confirmed that certain elements of our approach represent genuine contributions worth formalising. We accept the large majority of its recommendations and have proposed concrete implementation plans for each.

The most important insight from this exchange is not any single recommendation but the recognition that our pipeline sits at the intersection of multiple well-developed research traditions. We did not know, when we built the system, that our concept cards were implementing thesaurus term records, that our prerequisite graph was approaching a knowledge space, or that our card-to-guide architecture was mirroring cognitive architectural distinctions established in the 1980s. Knowing these things now does not change what the system does, but it fundamentally changes our ability to reason about it, evaluate it, and improve it.

We close with an observation about the nature of this exchange itself. The pipeline was developed through iterative conversation between a human practitioner and an LLM. The theoretical assessment was conducted through a similar process. The response and proposed improvements were developed in the same way. This entire arc—from pragmatic construction through theoretical critique to principled revision—has been conducted as a form of **LLM-mediated knowledge engineering**. If the pipeline is a first approximation of how to consolidate domain knowledge with LLM assistance, then this exchange is a first approximation of how to *validate and improve* such consolidation through the same medium. We note this not as a claim but as an observation that may itself warrant investigation.

References

Anderson, J.R. (1983). The Architecture of Cognition. Harvard University Press.

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

Doignon, J.-P. & Falmagne, J.-C. (1999). Knowledge Spaces. Springer-Verlag.

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

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

Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997). “METHONTOLOGY: From Ontological Art Towards Ontological Engineering.” AAAI Spring Symposium.

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

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

Guarino, N. & Welty, C. (2002). “Evaluating Ontological Decisions with OntoClean.” Communications of the ACM, 45(2), 61–65.

Guarino, N. & Welty, C. (2004). “An Overview of OntoClean.” In Handbook on Ontologies, Springer, 151–171.

Huang, L., et al. (2024). “A Survey on Hallucination in Large Language Models.” ACM Computing Surveys.

McGreggor, D.J. (2026). “From Source to Structure: A Practitioner’s Pipeline for LLM-Mediated Knowledge Consolidation.” Working Paper.

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

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

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

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

Poveda-Villalón, M., Gómez-Pérez, A., & Suárez-Figueroa, M.C. (2014). “OOPS! (OntOlogy Pitfall Scanner!).” IJSWIS, 10(2), 7–34.

Reigeluth, C.M. (1983). “The Elaboration Theory of Instruction.” In Instructional Design Theories and Models. Lawrence Erlbaum.

Ren, Y., et al. (2014). “Towards Competency Question-driven Ontology Authoring.” ESWC 2014.

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

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

Suárez-Figueroa, M.C., Gómez-Pérez, A., & Fernández-López, M. (2015). “The NeOn Methodology Framework.” Applied Ontology, 10(2), 107–145.

Tam, D., et al. (2024). “Let Me Speak Freely? A Study on the Impact of Format Restrictions on Performance of LLMs.” arXiv.

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.