From Source to Structure:
A Practitioner's Pipeline for LLM-Mediated Knowledge Consolidation

Duncan J. McGreggor

Working Paper — Draft for Discussion
February 2026

Abstract

This paper describes a multi-stage pipeline for transforming primary source material—textbooks, handbooks, academic papers—into a structured, queryable knowledge base suitable for both human reference and large language model (LLM) consumption. The pipeline was developed iteratively through practical experimentation rather than from first principles, driven by the concrete constraint of producing knowledge representations that an LLM could reliably access, process, and reason over. We present the pipeline descriptively, identify the implicit design decisions and epistemological commitments it encodes, and propose a preliminary framework for understanding these decisions as a first approximation of best practices for LLM-mediated knowledge consolidation. This is an early-stage report on a working system; we offer it not as a definitive methodology but as a documented starting point for further investigation, iteration, and peer evaluation.

1. Introduction

The problem of transforming unstructured or semi-structured knowledge into representations that support efficient retrieval, reasoning, and application is well-studied across several disciplines, including knowledge engineering, ontology design, information extraction, and library science. The emergence of large language models as both consumers and producers of knowledge introduces new constraints and opportunities that existing frameworks do not fully address.

This paper reports on a pipeline that was developed to meet a specific practical need: consolidating domain knowledge from authoritative primary sources into a form that an LLM could reliably access during interactive sessions. The system was not designed from theoretical first principles. Rather, it emerged through iterative experimentation—a series of conversations, tests, and refinements aimed at answering a deceptively simple question: what representation of

domain knowledge allows an LLM to give the most accurate, well-grounded, and useful responses?

The pipeline that emerged from this process transforms published source material through a sequence of stages, each producing artifacts at increasing levels of abstraction while maintaining provenance links back to the original sources. The system has been applied to music theory textbooks as an initial domain, producing a knowledge base of several hundred concept cards, dozens of synthesised guides, and a navigable knowledge graph—all served through a Model Context Protocol (MCP) server that provides structured access to the knowledge base during LLM interactions.

We present this work with an awareness of its limitations. The pipeline has been tested on a narrow set of domains. The design decisions that produced good results may reflect properties specific to those domains, to the particular LLM used, or to the specific use cases explored. Our purpose in documenting the pipeline at this stage is not to claim generality but to make the approach visible and available for critique, replication, and extension by others working on similar problems.

2. Pipeline Overview

The pipeline consists of four principal stages, each performing a distinct transformation on the source material. Figure 1 presents the complete pipeline architecture. The stages are: (1) source acquisition and format conversion, (2) concept extraction and categorisation, (3) synthesis and structural elaboration, and (4) unified indexing for retrieval. Stages 3 and 4 operate in parallel, with the synthesis layer producing guides and best practice documents while the structural layer constructs a formal knowledge graph. Both layers feed into a unified full-text search index that provides a single retrieval surface across all levels of abstraction.

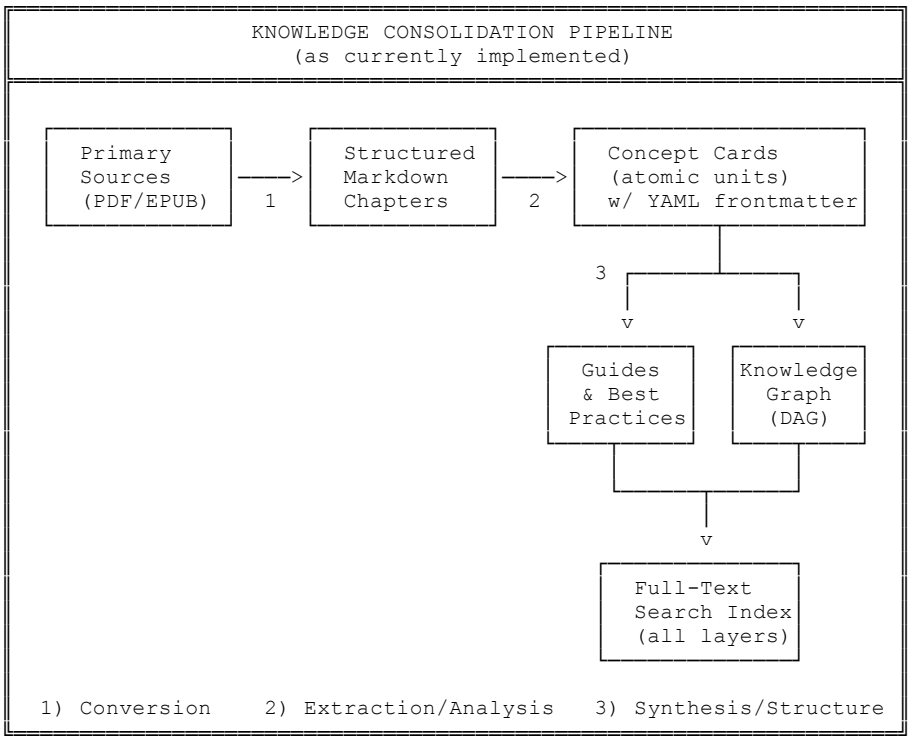

Figure 1. The knowledge consolidation pipeline, showing the four principal stages and the flow of artifacts between them.

3. Stage Descriptions

3.1 Stage 1: Source Acquisition and Conversion

The pipeline begins with primary source material in publication formats: PDF, EPUB, or LaTeX. These sources are converted to structured Markdown, with each logical division of the source—typically a chapter or major section—rendered as a separate file. Figure 2 illustrates the input-output relationship of this stage.

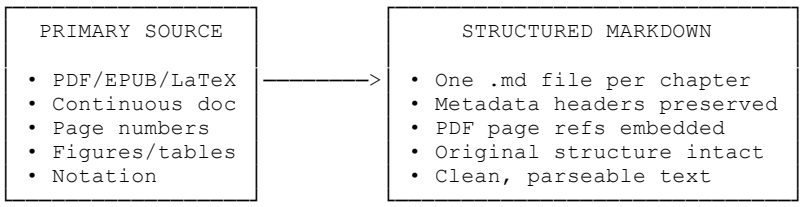

Figure 2. Stage 1 transformation: primary source to structured Markdown.

This conversion is not merely a format translation. It performs the first act of structuring by imposing a file-per-division organisation that creates natural units for subsequent parallel processing. Critically, metadata from the original source—including PDF page numbers and original line references—is preserved in embedded headers within each Markdown file, maintaining traceability to the physical source throughout all downstream transformations.

The choice of Markdown as the intermediate representation reflects a pragmatic optimisation: Markdown is natively legible to both humans and language models, requires no special parsing infrastructure, and preserves enough structural information (headings, lists, emphasis, code blocks) to support downstream processing without imposing the overhead of a richer markup language.

3.2 Stage 2: Concept Extraction and Categorisation

The structured Markdown chapters are analysed to identify discrete, atomic concepts. Each concept is extracted into a standalone artifact—a concept card—with a dual structure: machine-readable metadata encoded as YAML frontmatter, and human-readable content organised into standardised Markdown sections. Figure 3 illustrates the one-to-many decomposition that characterises this stage.

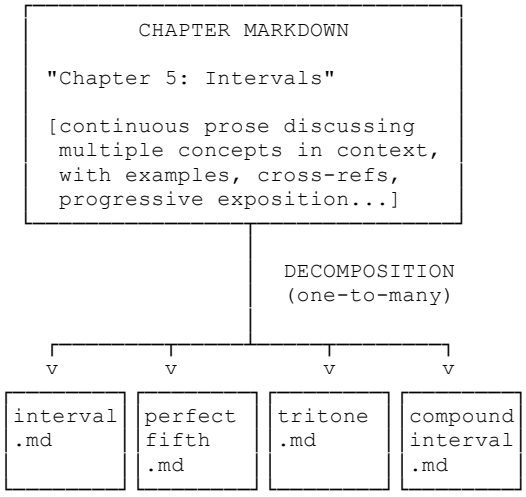

Figure 3. Decomposition of a source chapter into atomic concept cards.

The internal structure of each concept card is designed to serve multiple purposes simultaneously. Figure 4 presents the card anatomy. The YAML frontmatter constitutes a relational schema: the category field places the concept within a taxonomy, the source and chapter fields maintain provenance chains, and the chapter_number and pdf_page fields enable cross-referencing with the physical source. The Markdown content sections provide graded definitions (from accessible to formal), constructive or recognitional procedures, contextual information, source-specific examples, explicit cross-references to related concepts, and documentation of common misconceptions.

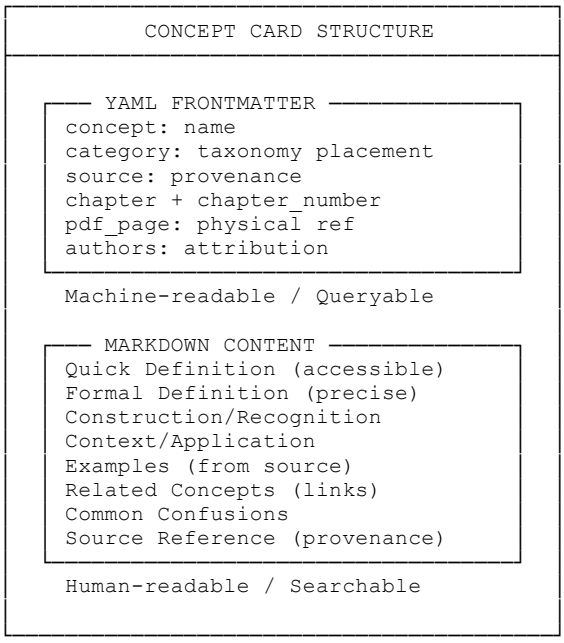

Figure 4. Anatomy of a concept card, showing the dual machine-readable and human-readable structure.

An important observation about this structure is that the "Related Concepts" section within the Markdown content body encodes informal graph edges—relationships between concepts that are expressed in natural language but can subsequently be formalised into typed edges in the knowledge graph. The concept card thus functions simultaneously as a standalone reference document and as a node pre-wired for integration into a network structure.

3.3 Stage 3a: Guide and Best Practice Synthesis

The concept cards produced in Stage 2 represent declarative knowledge—facts about individual concepts. Stage 3a transforms this declarative knowledge into procedural and heuristic knowledge by composing concept cards into synthesised guides. These guides do not merely catalogue concepts; they describe how concepts combine in practice, identify common workflows, articulate decision heuristics, and document idiomatic usage patterns. Figure 5 illustrates this many-to-many composition.

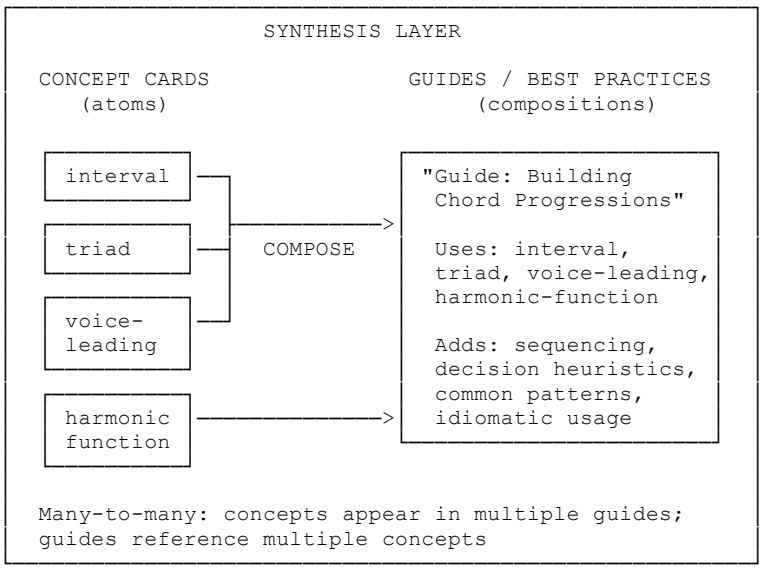

Figure 5. The synthesis layer: composing atomic concept cards into practical guides.

The distinction between concept cards and guides maps approximately onto the declarative–procedural knowledge distinction that has been extensively studied in cognitive science and knowledge engineering. Concept cards answer "what is X?"; guides answer "how do I use X, Y, and Z together to accomplish a goal?" This layering is significant because it mirrors the way domain expertise is typically structured: practitioners need both rapid lookup of individual facts and integrated understanding of how those facts compose into practice.

3.4 Stage 3b: Knowledge Graph Construction

In parallel with guide synthesis, the relationships between concepts—encoded informally in the "Related Concepts" sections and formally in the YAML metadata—are extracted and

materialised into an explicit graph data structure. This is not a metaphorical graph or a loose network of associations; it is a directed graph with typed edges, supporting standard graph-theoretic algorithms. Figure 6 illustrates a representative subgraph.

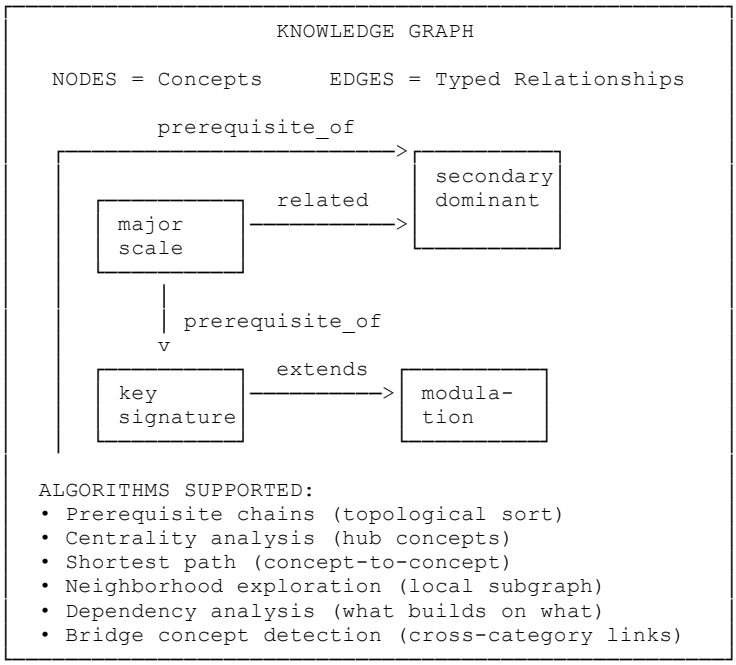

Figure 6. A representative subgraph from the knowledge graph, showing typed relationships and the algorithms they support.

The graph supports several classes of queries that would be difficult or impossible to answer from concept cards alone. Prerequisite chains (computed via topological sort) reveal the learning order for a given concept. Centrality analysis identifies hub concepts—those with disproportionately many connections—which often correspond to foundational ideas in the domain. Shortest-path computation between concepts can reveal non-obvious connections. Neighbourhood exploration provides local context around any concept. Bridge concept detection identifies concepts that span multiple categories, often representing points where distinct subdomains intersect.

3.5 Stage 4: Full-Text Search Indexing

The final stage indexes all artifacts from every preceding layer—source chapters, concept cards, unified (cross-source) concepts, and guides—into a single full-text search index with relevance ranking. Figure 7 illustrates the indexed content types.

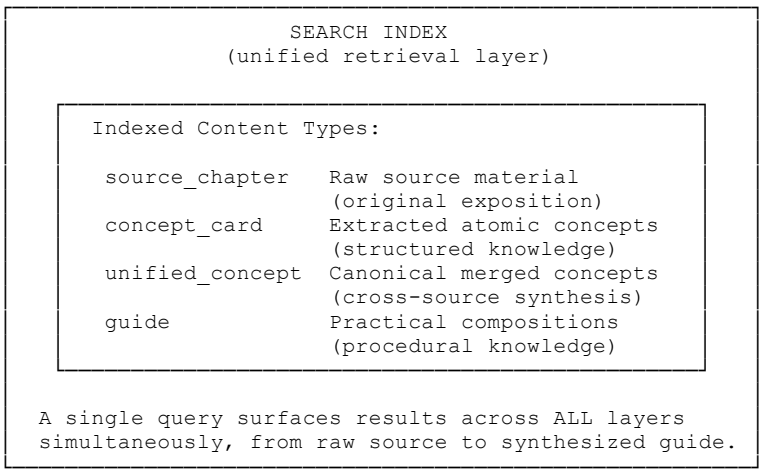

Figure 7. The unified search index, providing a single retrieval surface across all abstraction levels.

This unified indexing is significant because it allows a single query to surface results at multiple levels of abstraction simultaneously. A query for a technical term might return the original source exposition (Level 0), the extracted concept card (Level 1), a cross-source unified concept (Level 2), and a practical guide that uses the concept (Level 3). This multi-level retrieval supports different information needs within a single interaction: quick lookup, deep understanding, cross-referencing, and practical application.

4. Structural Observations

4.1 Abstraction with Provenance

A structural property of the pipeline that warrants explicit identification is the relationship between abstraction levels. Figure 8 presents the five levels of abstraction produced by the pipeline.

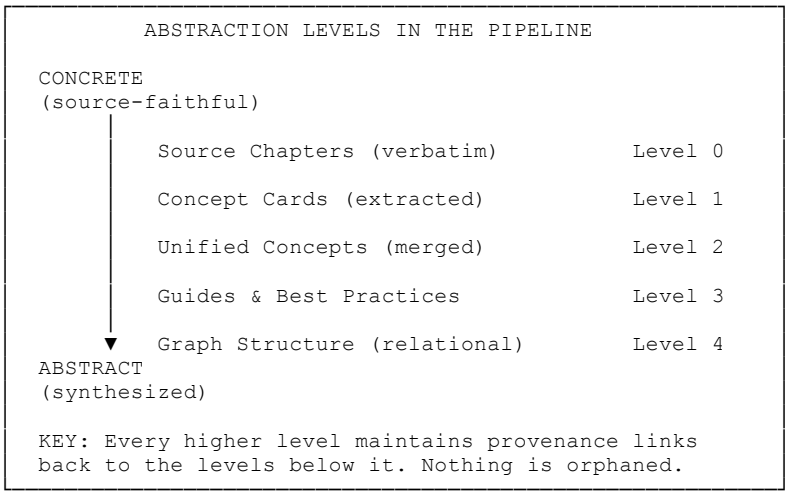

Figure 8. The five levels of abstraction in the pipeline, from concrete source material to abstract relational structure.

The pipeline moves from concrete to abstract through progressive transformation, but critically, it never severs the provenance chain. A guide references the concept cards it composes. Each concept card references the source chapter and page from which it was extracted. The source chapter references the physical page in the original publication. At every level, it is possible to trace an assertion or a relationship back to its origin in the primary source material. This property—which we term **provenance-preserving abstraction**—was not designed as a theoretical commitment. It emerged from the practical requirement that the LLM be able to verify and ground its responses by reference to authoritative sources. Nevertheless, it aligns with established principles in knowledge provenance and data lineage research.

4.2 Dual Representation

Each concept card encodes the same knowledge in two complementary formats: a machine-readable YAML schema and a human-readable Markdown exposition. This dual representation serves distinct functions. The YAML frontmatter supports programmatic operations: filtering, sorting, taxonomy navigation, and graph construction. The Markdown content supports retrieval, presentation, and natural-language reasoning. Neither representation is redundant; they address different access patterns and are consumed by different downstream processes.

This duality was, again, a pragmatic decision rather than a principled one. Early iterations of the system used either pure structured data (which was precise but difficult for the LLM to present conversationally) or pure prose (which was readable but difficult to query systematically). The hybrid format emerged as a compromise that served both needs. We note, however, that this compromise has parallels in knowledge representation research, where the tension between formal and informal representations has been a persistent theme.

4.3 Declarative–Procedural Layering

The separation between concept cards (declarative) and guides (procedural) reflects a distinction that is well-established in both cognitive science and knowledge engineering. Concept cards capture **knowing-that**: the facts, definitions, properties, and relationships of individual concepts. Guides capture **knowing-how**: the workflows, heuristics, patterns, and decision procedures by which concepts are applied in practice. This layering was not initially planned; it arose because the concept cards alone, while excellent for lookup, did not adequately support task-oriented queries such as "how do I harmonise a melody?" or "what is the standard approach to modulation?" The addition of the guide layer addressed this gap.

4.4 Graph as First-Class Structure

The decision to materialise concept relationships as a literal graph data structure—rather than relying on embedding-based similarity or purely textual cross-references—has significant implications. A materialised graph supports algorithmic reasoning: topological sorting for prerequisite ordering, centrality computation for identifying foundational concepts, shortest-path calculation for finding connections between distant concepts, and bridge detection for cross-domain integration. These operations would be difficult to perform reliably on embedding-based or purely textual representations. The graph also provides a form of structural validation: orphaned concepts (those with no edges) and cycles in prerequisite chains (which would indicate logical inconsistencies) can be detected automatically.

5. Limitations and Future Work

This report describes a working system, not a validated methodology. Several significant limitations should be noted.

Domain breadth. The pipeline has been applied primarily to music theory textbooks. While the architecture is designed to be domain-agnostic—with customisable card templates and category taxonomies—its effectiveness in substantially different domains (e.g., experimental sciences, legal texts, software documentation) has not been systematically evaluated.

Evaluation methodology. Quality assessment of the extracted knowledge has relied on expert review of sampled concept cards and functional testing of the retrieval system. No formal evaluation against established knowledge engineering benchmarks has been conducted.

LLM dependency. Both the extraction process and the downstream consumption of the knowledge base are mediated by large language models. The design decisions that produced good results may be entangled with properties of specific models (particularly their strengths and limitations in processing structured text). The degree to which this pipeline reflects general principles of knowledge organisation versus LLM-specific optimisations remains an open question.

Feedback loops. The current pipeline is largely unidirectional. Although the graph construction stage occasionally reveals gaps in concept extraction (e.g., orphaned nodes that suggest missing prerequisite cards), there is no formalised feedback mechanism by which later stages systematically inform earlier ones. Establishing such feedback loops is a priority for future development.

Comparison with existing frameworks. This work has not yet been systematically compared with established approaches in ontology engineering (e.g., Methontology, NeOn), knowledge graph construction (e.g., knowledge graph embedding methods), or curriculum sequencing (e.g., prerequisite structure learning). Such comparison is a natural and necessary next step.

We view the pipeline described here as a first approximation—a pragmatically-derived starting point that has demonstrated utility in its initial application. The immediate priorities for further work include: extending the pipeline to additional domains, developing formal evaluation

criteria, investigating feedback mechanisms between pipeline stages, and situating the approach within the existing literature on knowledge representation and transformation. We offer this report as an invitation to that broader investigation.