# **Technical Research Report: A Professional-Grade Agentic System with Tool-Augmented Auditing and Structured State**

Version: 18.0

Date: 15 June 2025

### 1. Abstract

This document details the final proposed evolution of the agentic architecture before implementation, advancing the v17 Teacher-Student model to a professional-grade standard. This is achieved by incorporating two industry best practices: Tool-Augmented Auditing and Structured JSON State Exchange. The previous architecture, while robust, still relied on the Teacher/Auditor LLM to compare unstructured text for fact-checking, a process susceptible to interpretation errors. The v18 model addresses this by equipping the Auditor agent with a dedicated citation\_retriever tool, forcing it to programmatically fetch and verify claims against the source documents. This transforms the audit from a qualitative review into a verifiable, programmatic process. Furthermore, to eliminate the brittleness of passing state via Markdown files, all inter-graph communication is upgraded to use structured JSON objects. The Planning graph now produces a plan.json and the Evaluation graph produces a detailed evaluation.json. This ensures that the Supervisor's control flow is governed by reliable, machine-readable data, not fragile string parsing. These enhancements represent the final step in creating a truly robust, verifiable, and scalable agentic research system.

### 2. Research: From Qualitative Review to Programmatic Verification

The v17 architecture established a crucial Teacher-Student dynamic with an independent auditor. However, the audit process itself remained qualitative. The Teacher LLM was asked to *read* a synthesized claim and *read* a source text, then form an *opinion* on whether they matched. This still leaves room for error and hallucination within the audit process itself.

Industry-leading agentic systems (e.g., Google's internal tooling, advanced ReAct agents) mitigate this by forcing the agent to use tools. The LLM's primary task shifts from "knowing the answer" to "knowing how to use a tool to find the answer."

* Tool-Augmented Auditing: By providing the Teacher agent with a citation\_retriever tool, we change its task fundamentally. To verify a claim, it *must* formulate a precise query for the tool, execute it against the knowledge base, and then perform a simple comparison on the returned data. This grounds the audit in a programmatic action, making it far more reliable and transparent.
* Structured State: Complex software systems do not communicate by writing reports to each other; they use APIs that pass structured data. Our agent system should be no different. Relying on the agent to perfectly format Markdown, and then relying on the supervisor to perfectly parse it, is a major potential failure point. Migrating all state exchange (plans, evaluations) to JSON eliminates this entire class of errors.

### 3. The v18 Architecture: A Production-Ready Framework

The v18 architecture retains the tri-phasic Teacher-Student workflow of v17, but upgrades the implementation details to professional standards.

* Student (Gemini):
  + Planning Phase: Now generates a plan.json file. This JSON object will contain a list of structured plan items, each with a title and a detailed description.
  + Synthesis Phase: Unchanged, but still required to embed machine-readable citation tags in its Markdown output.
* Teacher/Auditor (DeepSeek with Tools):
  + Evaluation Phase: This graph is now tool-enabled.
    1. It first parses the synthesized report to extract all claims and their associated citation tags.
    2. For each claim, it uses the citation\_retriever tool to fetch the exact text from the source document specified in the tag.
    3. It then uses the LLM to perform a direct comparison between the claim and the retrieved text, outputting a PASS or FAIL status.
    4. It generates its final evaluation.json object, which includes a structured list of all citation checks, along with its qualitative review of consistency and goal alignment.
* Supervisor (supervisor.py):
  + The supervisor's logic is now cleaner and more robust. It parses JSON objects from the agent outputs, not Markdown.
  + It presents the plan.json to the user in a human-readable format.
  + It provides the evaluation.json to the user in a clear, structured report format (e.g., a table of citation checks).

This architecture represents a mature, robust, and verifiable system that aligns with the best practices for building reliable AI agents. It provides the strongest possible foundation for our coding and experimentation phase.