# **Consolidated Technical Research Report**

Version: 1.0 (Synthesizing v2-v19)

Status: Canonical

## **1. Abstract**

This report documents the complete conceptual evolution of a professional-grade, agentic research system. The project's journey began with a standard Retrieval-Augmented Generation (RAG) model and systematically progressed through increasingly sophisticated architectures to solve key challenges in agentic AI. Core architectural shifts include the transition from a single agent to a multi-agent "constitutional committee," the introduction of a stateful supervisor to manage context, the formalization of a Plan-Synthesize-Evaluate workflow with Human-in-the-Loop (HITL) gates, and a pivotal simplification to a Teacher-Student model for robust, independent auditing. The final, stable architecture achieves verifiability and reliability through two primary innovations: Tool-Augmented Auditing, which forces the evaluator agent to programmatically verify claims against source documents, and a simplified, single-call evaluation graph, which enhances robustness by entrusting the entire audit process to a single, guided LLM call. This narrative captures the key research findings and architectural decisions that led to the current stable, iterative, and accountable system.

## **2. Foundational Architectures (v2-v6)**

The project's inception was a Personalized AI Research Agent (v2) built on a standard LangChain RAG architecture. Its goal was to answer questions based on a user-provided knowledge corpus. Early development quickly revealed the limitations of simple RAG for complex tasks, leading to the adoption of LangGraph to create a stateful, cyclical agent (v3). This introduced the first core agentic loop: generate-test-reflect-regenerate, specifically for code generation.

This was then generalized into a multi-path system (v4) capable of routing between "research" and "code" generation tasks. To improve quality, a generate -> reflect -> refine loop was added to the research path (v5), and to improve debuggability, chain-of-thought logging was introduced to make the agent's internal reasoning observable (v6).

## **3. The Multi-Agent System & The Constitution (v7-v9)**

To address the problem of self-correction bias, the architecture pivoted to a multi-agent system (v7). The initial design involved two identical agents from different LLM families critiquing each other's work. However, this led to unaligned, destructive criticism.

The solution was the introduction of a Shared System Prompt, or "Constitution" (v8), which reframed the adversarial critique into a structured, collaborative debate. All agents were governed by the same set of principles and goals, forcing them to work together. This paradigm was then extended into a tri-modal system (v9) with dedicated workflows for Research, Code, and Documentation generation.

## **4. The State-Passing Supervisor & Formalized Workflow (v10-v15)**

While the constitutional system improved output, it suffered from context pollution. The Supervisor was enhanced into a Sequential, State-Passing Manager (v10) that explicitly passed the output of one stage as the primary input for the next, ensuring continuity. This was coupled with the externalization of prompts into a prompts.yaml file for modularity (v11).

A key logical flaw was identified: generating code directly from research was error-prone. The workflow was scientifically reordered to Research -> Technical Design Specification -> Code (v12), introducing a formal design phase that dramatically reduced logical errors.

To handle increasing complexity, the monolithic generation process was decomposed into a Plan, Synthesize, Review cognitive workflow (v13), which was later formalized under the general Perception-Reasoning-Action (PRA) framework (v14). Finally, to ensure user alignment, a critical HITL gate was introduced with Supervised Intent-Planning (v15), requiring user approval of a generated plan before synthesis could begin.

## **5. The Teacher-Student Model & Tool-Augmented Auditing (v16-v18)**

The multi-agent "debate" committee, even with a constitution, proved inefficient and prone to cognitive echo chambers. The architecture underwent its most significant simplification and pivot to a Teacher-Student model (v17). This model designated a "Student" agent (e.g., Gemini) for all generation tasks and an independent "Teacher" agent from a different model family for evaluation. The process was no longer a debate, but a formal, independent audit.

To perfect this audit, the system adopted two professional-grade standards (v18):

1. Tool-Augmented Auditing: The Teacher agent was equipped with a citation\_retriever tool, forcing it to programmatically verify the Student's claims against the source documents. This moved the audit from a qualitative opinion to a verifiable, programmatic process.
2. Structured JSON State Exchange: Communication between graphs (e.g., plans and evaluations) was upgraded from fragile Markdown files to reliable JSON objects, improving system robustness.

## **6. Final Refinement: Single-Call Evaluation (v19)**

While the tool-augmented Teacher was a major step forward, early implementations using a multi-node evaluation graph proved brittle, suffering from state-passing errors. The final architectural refinement, reflected in the stable v2 codebase, was to collapse the entire evaluation process into a single, robust evaluation node.

This node is guided by one comprehensive prompt that instructs the LLM to perform the full reasoning chain internally: parse claims, use its bound tools to verify them, assess goal alignment, and synthesize all findings into a single JSON output. This approach leverages the advanced reasoning of modern LLMs to handle complex internal workflows, eliminating multiple points of failure in the graph itself and resulting in a more resilient and maintainable system. This represents the canonical architecture of the project.