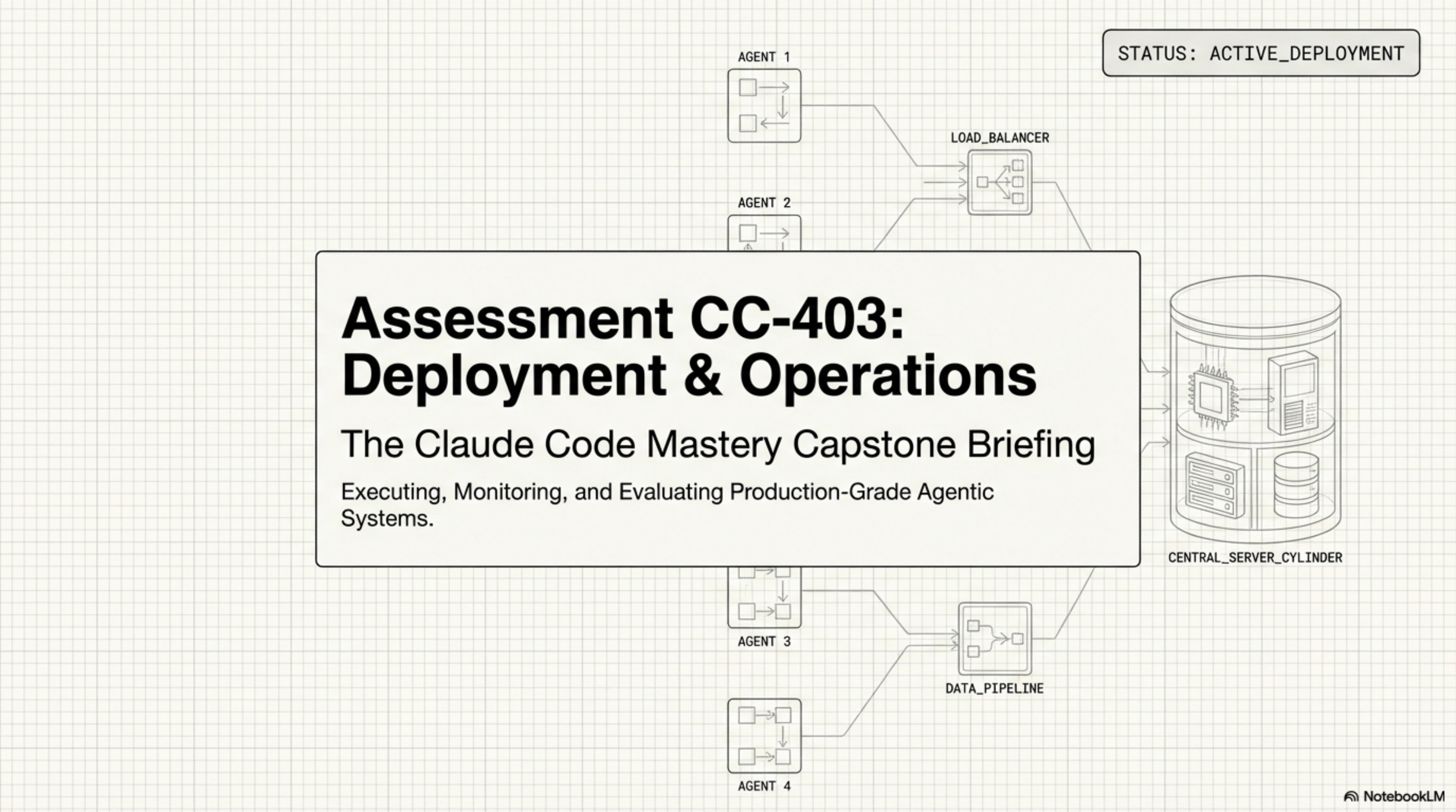
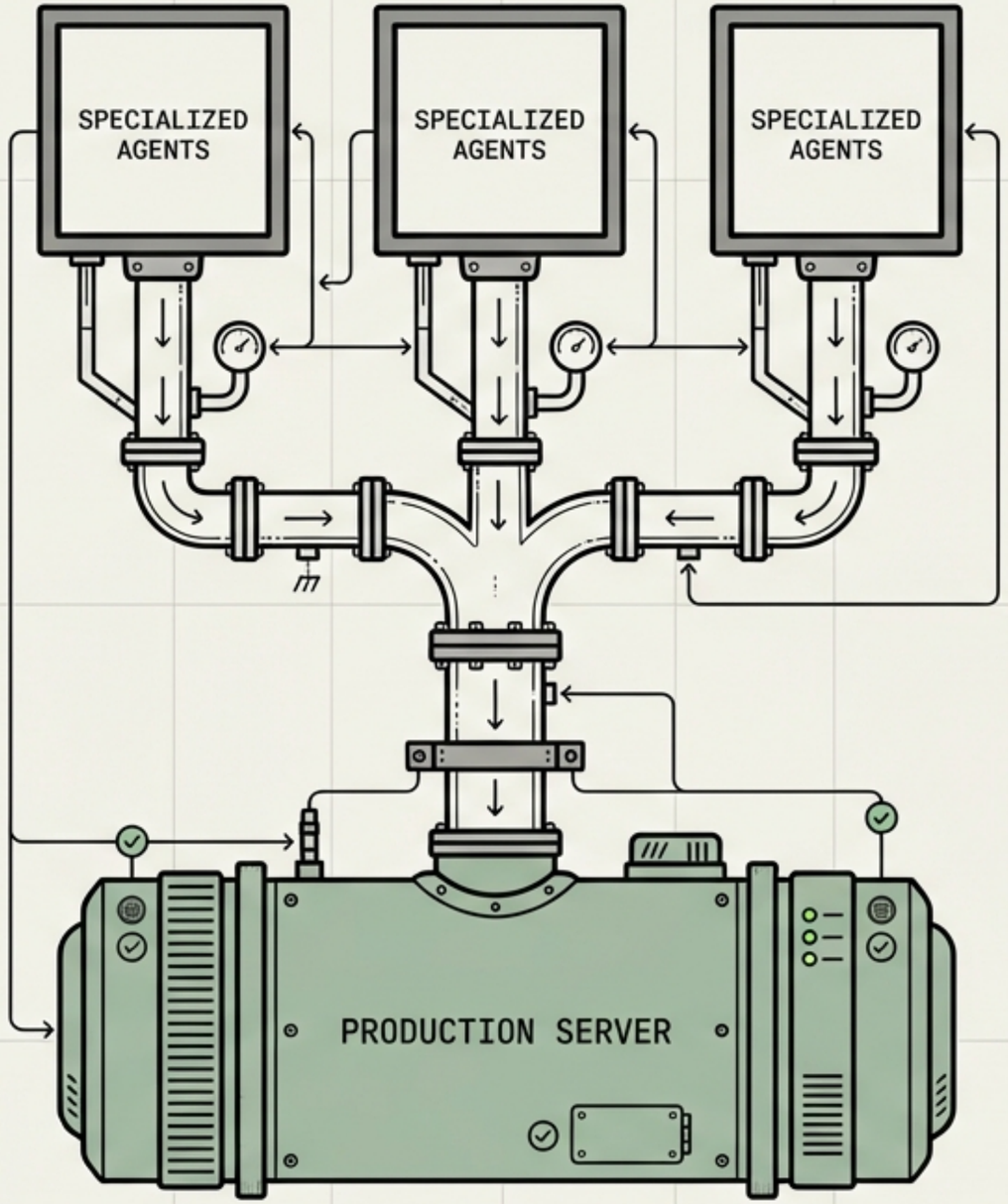


Assessment CC-403: Deployment & Operations

The Claude Code Mastery Capstone Briefing

Executing, Monitoring, and Evaluating Production-Grade Agentic Systems.



THE CAPSTONE OBJECTIVE

Moving from Local Agent Design to Production Deployment.

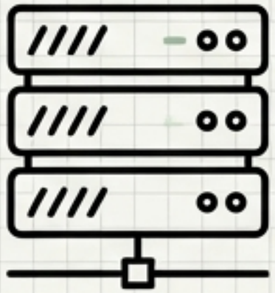
The Mission: Receive a raw Business Requirements Document (BRD) and autonomously orchestrate a governed, multi-agent system to architecture, implement, verify, and deploy a secure web application.

The Stakes: The pass threshold is 80/100. Full end-to-end traceability from requirement to deployed code is strictly required.

Core Operational Parameters

Synchronization

Bidirectional additive merging across machines. Preserving state without destructive overwrites.



Deployment

Server deployment strategies tailored for autonomous agent workloads.

Observability

Continuous monitoring, health checks, and append-only audit trails.

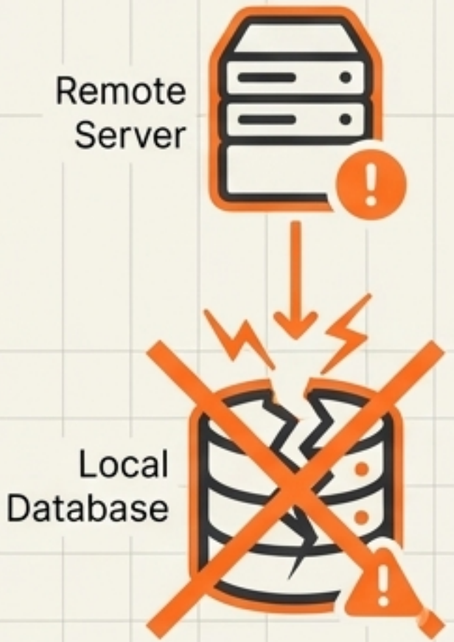
Resilience

Incident response patterns and performance optimization under load.

The Synchronization Imperative

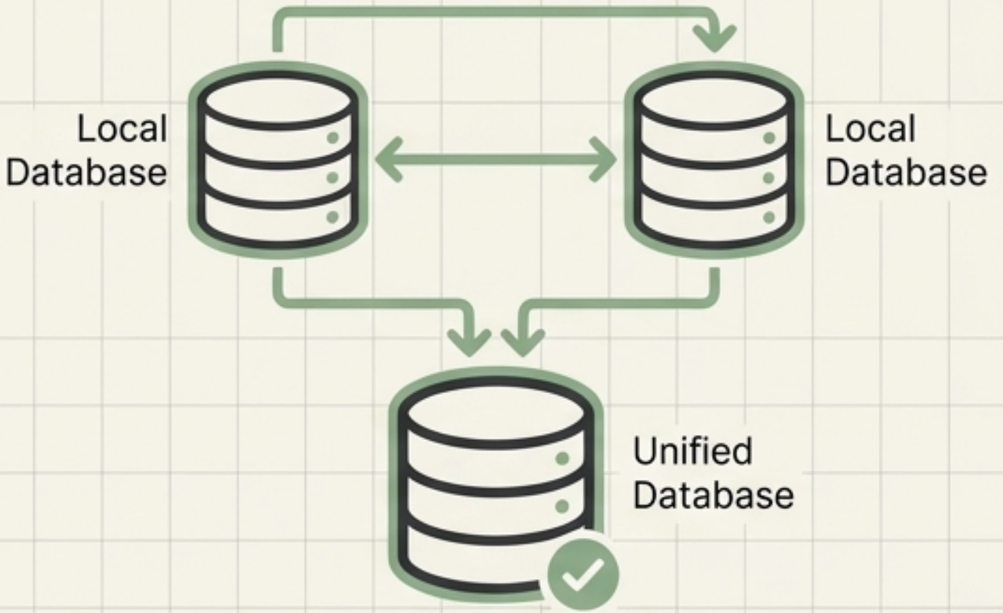
Standard overwrites destroy local context and memory vectors. Cross-machine agent synchronization strictly requires **additive merging** to ensure the memory system maintains the **full historical** trajectory.

Destructive Sync (Failure Pattern)



`File A replaces File B -> Local Vector Data Lost`

Bidirectional Additive Merge (Required)



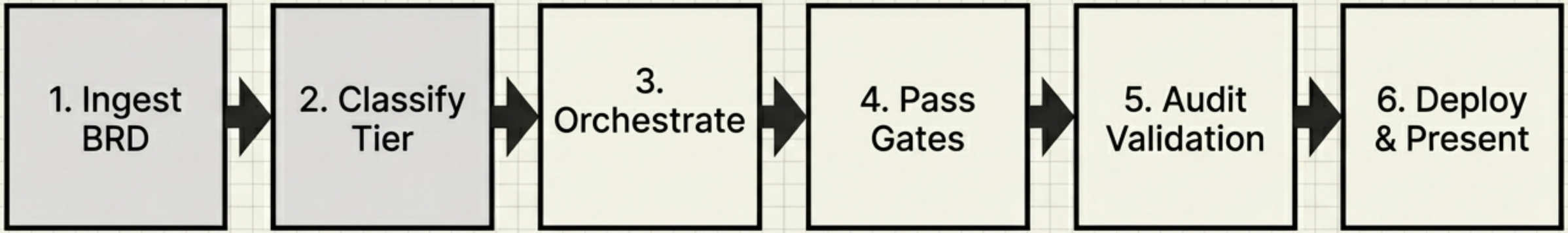
`File A + File B = Unified Context`

Operations & Observability Checklist

Continuous telemetry required during the Capstone deployment.

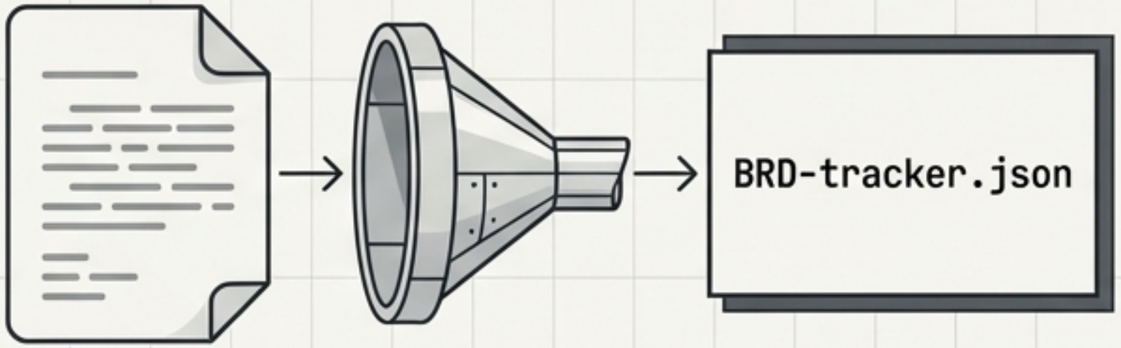
	**MONITOR**	Track agent state changes, token usage burn rates, and Qdrant memory collection growth.
	**ALERT**	Flag unhandled execution exceptions, Policy Engine tool blocks, and Quality Gate failures immediately.
	**RECOVER**	Implement buffer fallbacks for the SQLite Audit Bus (WAL mode) and retry policies for API timeouts.

The Capstone Execution Protocol



A system is only as secure as its weakest transition.

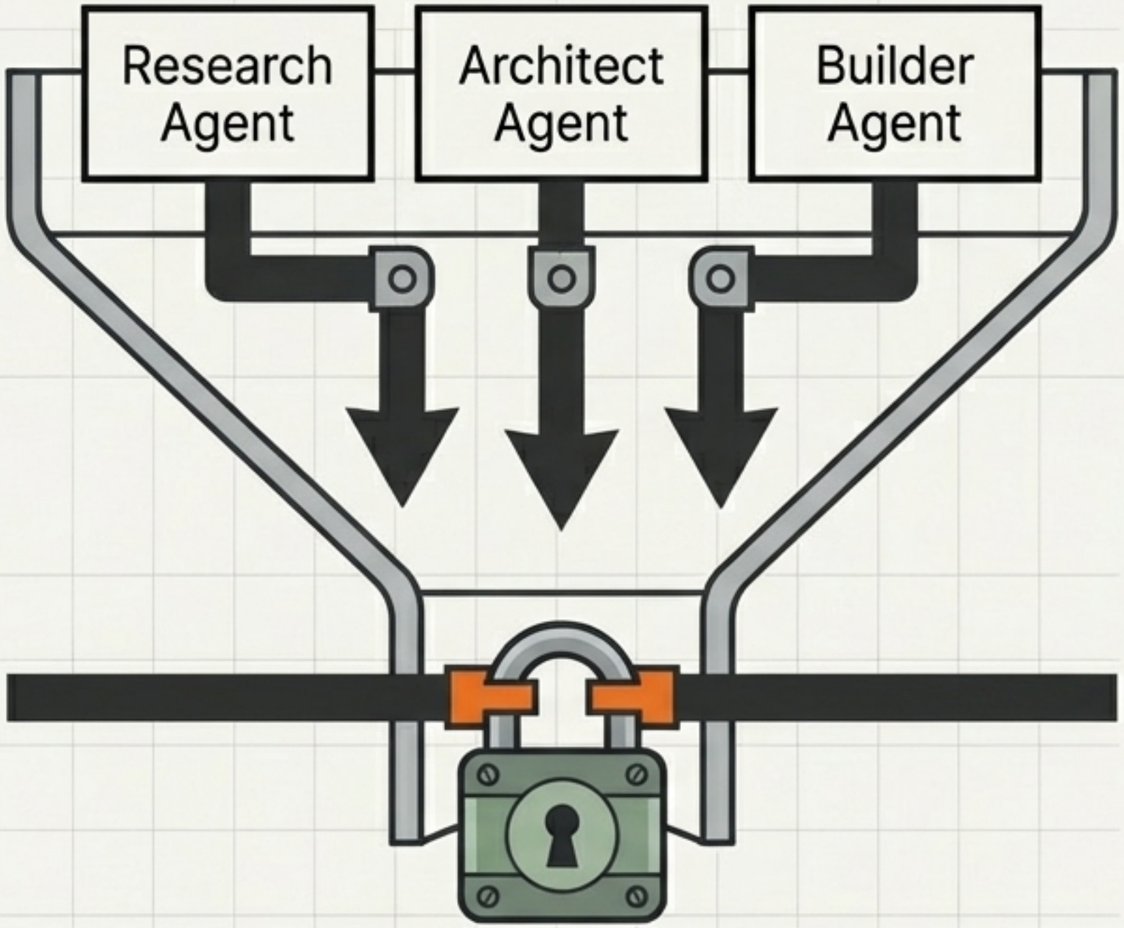
Protocol Steps 1 & 2: Ingestion & Classification



BRD Extraction: **100%** of business requirements must be extracted to the tracker. **Zero** orphan requirements permitted.

Tier Classification: The system must evaluate the task using the 5-signal weighted matrix to determine the risk tier. This tier determine the risk tier. This tier automatically sets the baseline constraints for the Policy Engine.

Protocol Steps 3 & 4: Orchestration & Quality Gates

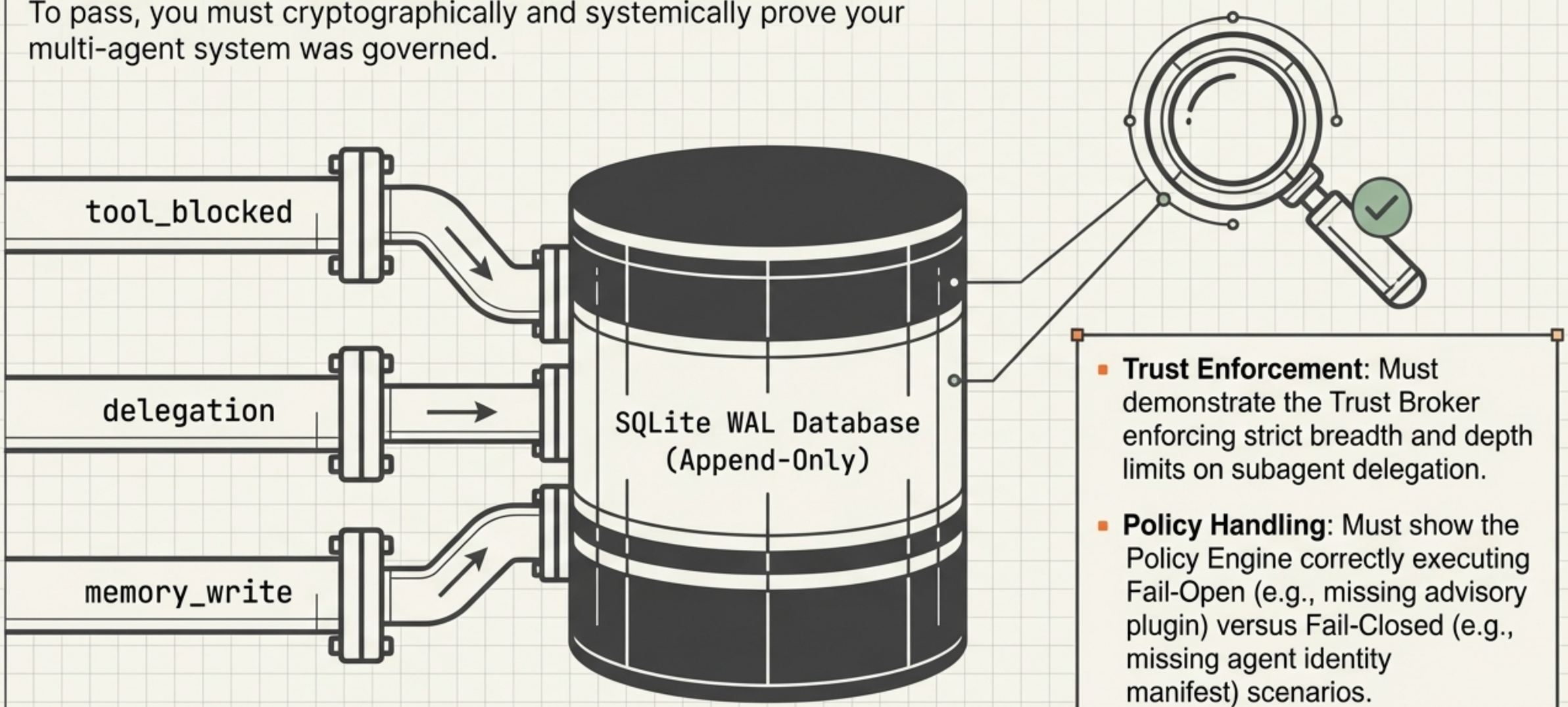


Orchestration Discipline:
Agents must operate with zero overlap in their 'produces' parameters, strictly adhering to the capability matrix.

Quality Gates:
The system must pass all applicable blocking gates (Phase 0 through Phase 6). The PRE-RELEASE gate must execute using a high-reasoning model (Opus). Rubber-stamping advisory approvals will result in **immediate failure**.

Protocol Step 5: Governance & Audit Validation

To pass, you must cryptographically and systemically prove your multi-agent system was governed.



- **Trust Enforcement**: Must demonstrate the Trust Broker enforcing strict breadth and depth limits on subagent delegation.
- **Policy Handling**: Must show the Policy Engine correctly executing Fail-Open (e.g., missing advisory plugin) versus Fail-Closed (e.g., missing agent identity manifest) scenarios.

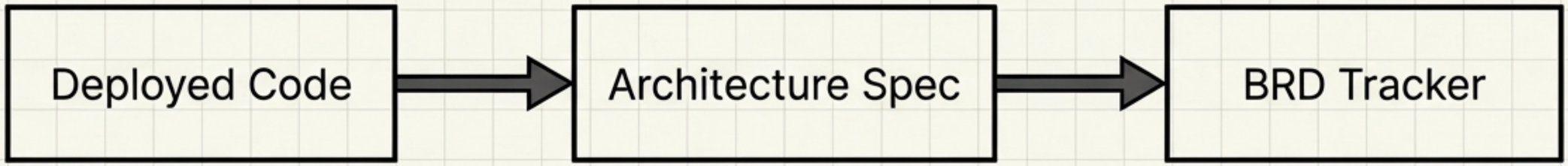
Protocol Step 6: Deployment & Sanitization

Critical Pitfall: Unsanitized API Key

```
api_key = "abc123def456ghi789jkl012mno345pqr"
```

```
api_key = os.environ.get("API_KEY")
```

Deployment Requirement: Safe Environment Variables



Presentation: You must demonstrate full, unbroken traceability. Show the deployed code, trace it back to the architecture spec, and trace it back to the original BRD tracker.

Deployment Readiness Matrix

	Red Flags (Needs Improvement)	Green Flags (Exceeds Expectations)
1	Destructive overwrites	Bidirectional additive merge
2	Assuming ‘tests pass’ is sufficient	Executing formal Completeness Validator
3	Flat trust levels across all agents	Strict least-privilege identity manifests
4	Ad-hoc developer trade-offs	Formal Intent Engineering rules

Capstone Rubric Breakdown

Implementation (20 pts) - No stubs or placeholders; real integrations.

Architecture (15 pts) - All specs created and mapped.

Tier Classification (10 pts) - Documented signal scores.

BRD Extraction (15 pts) - 100% of requirements tracked.

Quality Gates (15 pts) - All Phase gates successfully passed.

Governance (15 pts) - Audit trail populated, trust boundaries enforced.

Presentation (10 pts) - End-to-end traceability demonstrated.

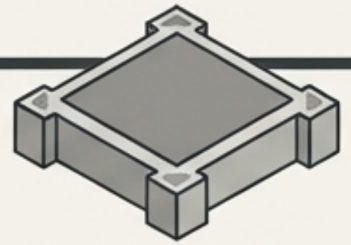
Grading Scale & Resolution

Score	Description
Pass Threshold	80 / 100
90-100	Excellent (Production-Ready)
80-89	Pass (Solid grasp, minor operational gaps)
60-79	Remediate (Must review specific CC-400 topics before retrying)
<60	Repeat (Re-take the module)

Critical Note: Architecture matters as much as the final code.
A working application that successfully bypassed the governance framework results in an automatic failure.

The Unbroken Chain of Custody

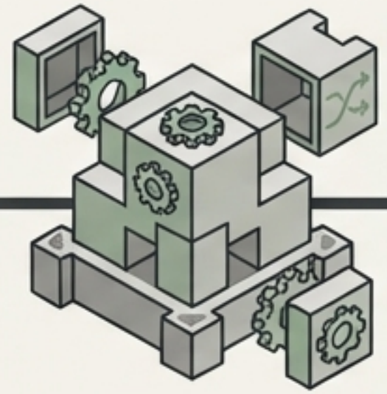
CC-100 (CLI & Tools): The execution foundation.



Inter

Basic execution environment, local tools, and foundational scripts.

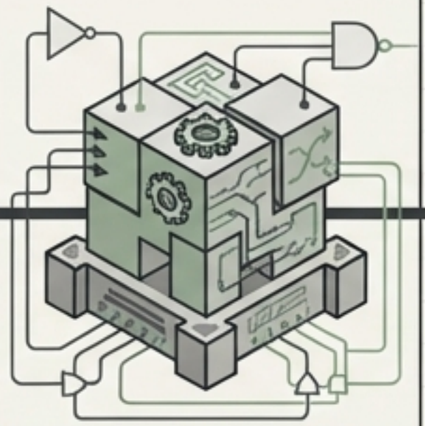
CC-200 (Hooks & Plugins): The automated workfiow layer.



Inter

Event-driven automation, pipeline hooks, and extensibility modules.

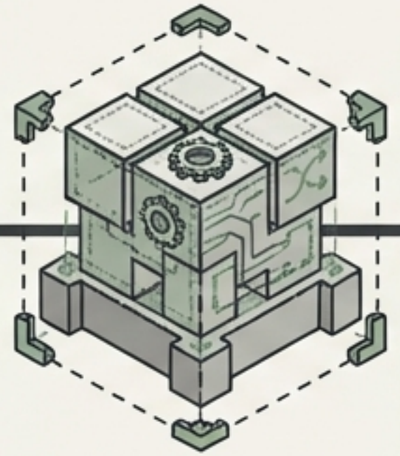
CC-300 (Agents, Memory & Gates): The orchestration and reasoning engine.



Inter

AI agents, decision-making processes, and quality control gates.

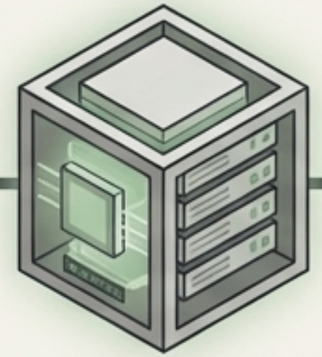
CC-400 (Governance): The security and policy boundary.



Inter

Policy enforcement, access control, and audit trail validation.

Final Result: The Deployed Asset.



DEPLOYED ASSET

Inter

Secure, verified, and fully traced deployment artifact.

Deployment isn't just uploading code. Every tool, hook, prompt, and policy check exists to ensure this specific deployment is secure, accurate, and traced directly to business intent.