Documentation for Automaton Auditor Swarm

Chapter 1: Dialectical Synthesis

Dialectical synthesis is defined as the process of combining thesis and antithesis to form a new coherent whole. In our system, it's implemented through the Justice node which merges findings from multiple detectives. For example, when two detectives disagree, the Justice node performs synthesis by analyzing both perspectives and generating a unified conclusion.

Chapter 2: Fan-in and Fan-out Architecture

Fan-in refers to the number of inputs a node receives. In our graph, the Detective nodes have high fan-in as they receive data from multiple sources. Fan-out is the number of outputs from a node - the Judge nodes broadcast decisions to all listeners. This pattern is implemented in src/graph.py where nodes have multiple edges.

Chapter 3: Metacognition in Agents

Our agents implement metacognition through self-reflection mechanisms. Each Detective can evaluate its own confidence levels and request second opinions. The code in src/nodes/detectives.py shows how metacognition is implemented with confidence thresholds and recursive self-evaluation.

Chapter 4: State Synchronization

State synchronization across distributed nodes is achieved through the StateManager class in src/state.py. It ensures all nodes have consistent views of the system state. The synchronization protocol uses vector clocks and version vectors to detect conflicts.

File Structure:

- src/state.py - State management
- src/graph.py - Graph architecture
- src/nodes/detectives.py - Detective nodes
- src/nodes/judges.py - Judge nodes

- src/nodes/justice.py - Justice node

- tests/test_detectives.py - Detective tests

- src/tools/repo_tools.py - Repository tools