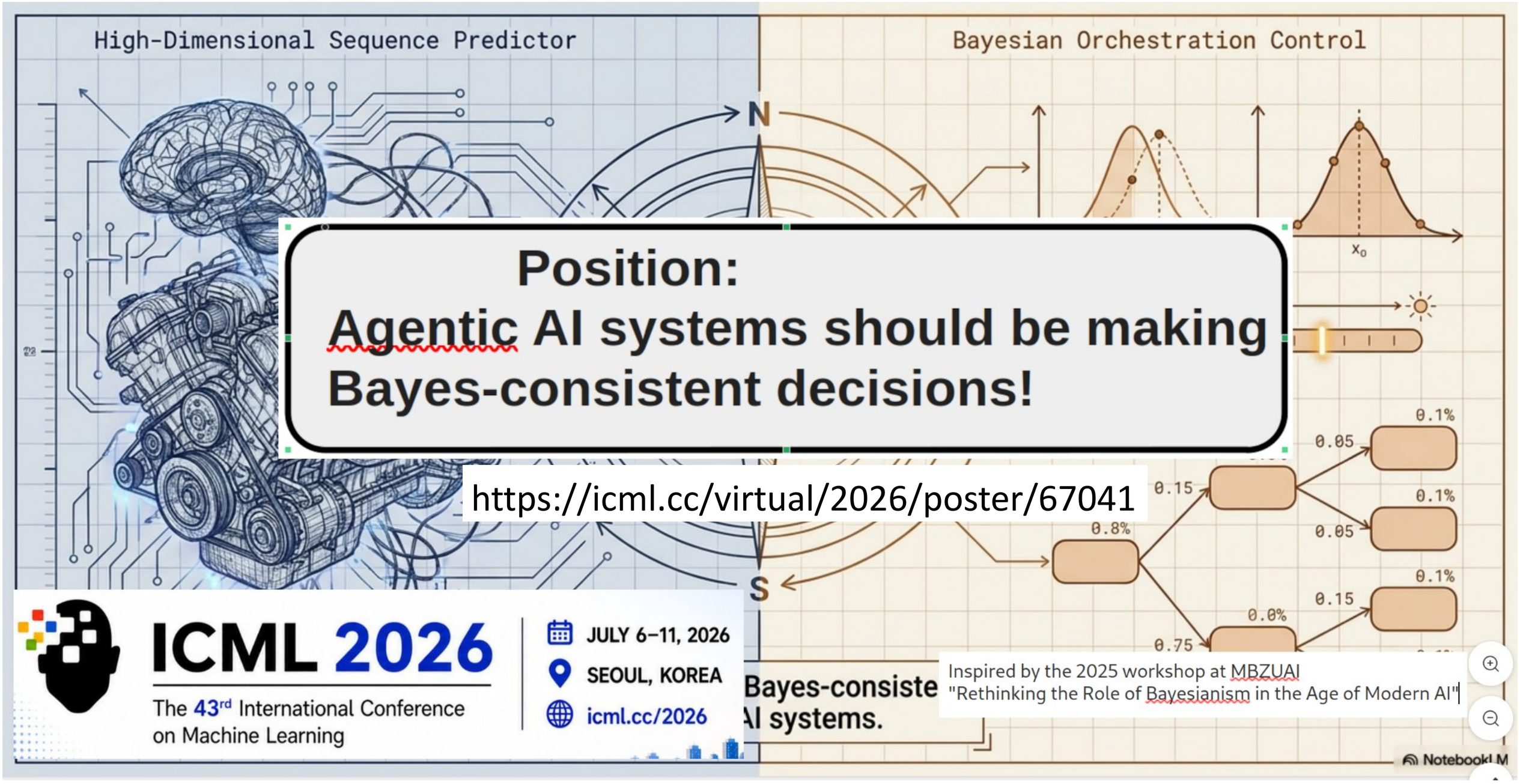
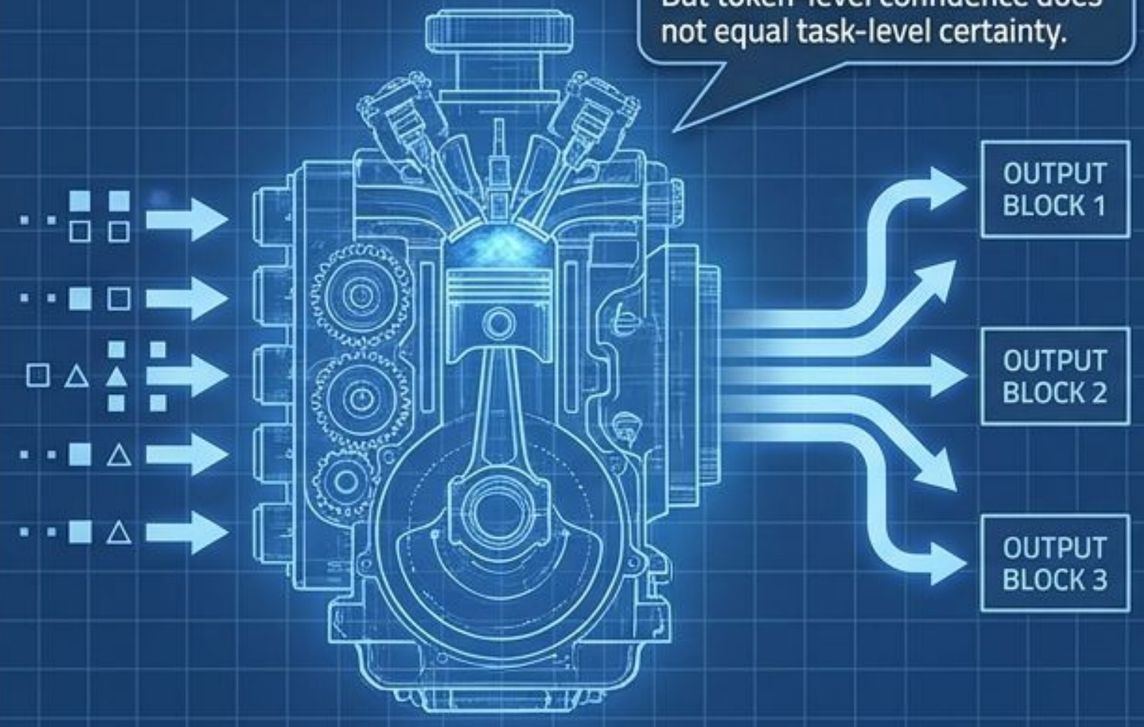
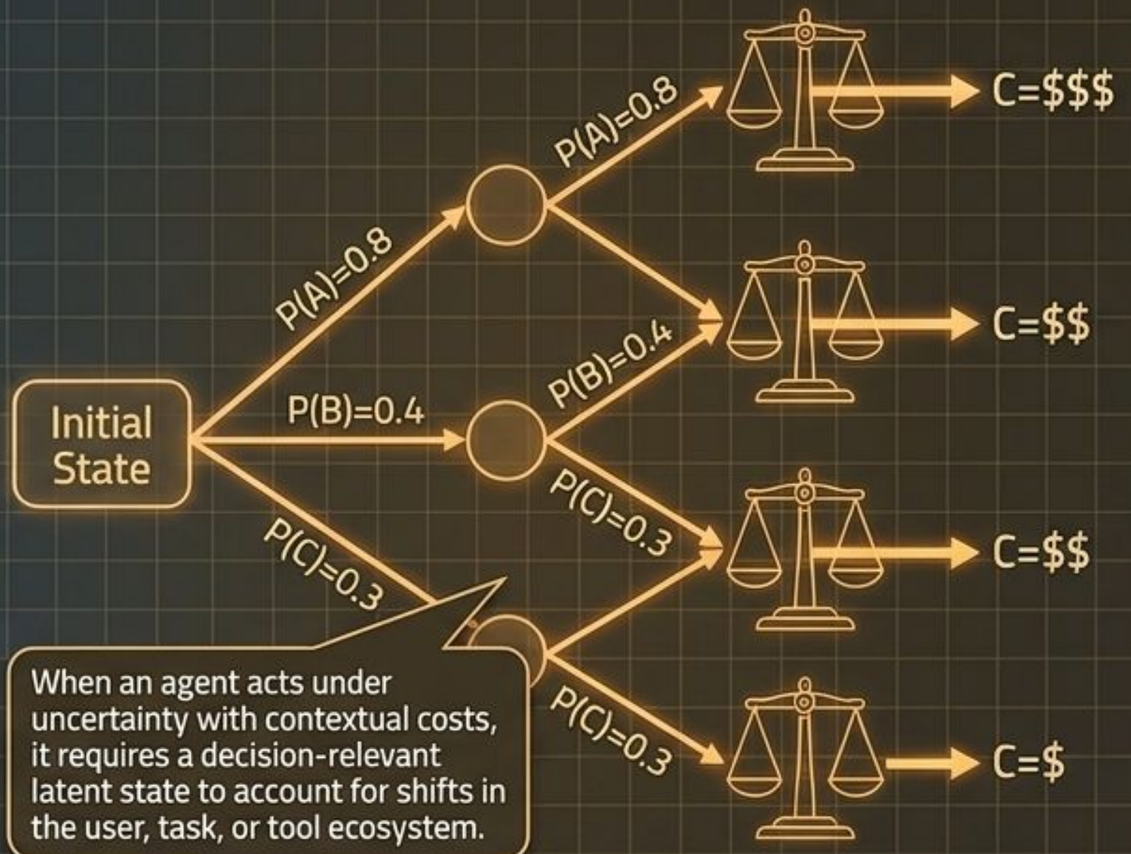

High-Dimensional Sequence Predictor
Bayesian Orchestration Control
N
S
x_0
0.1%
0.05
0.15
0.8%
0.1%
0.05
0.1%
0.15
0.0%
0.75
Position:
Agentic AI systems should be making Bayes-consistent decisions!
https://icml.cc/virtual/2026/poster/67041
ICML 2026
The 43rd International Conference on Machine Learning
JULY 6–11, 2026
SEOUL, KOREA
icml.cc/2026
Bayes-consiste
AI systems.
Inspired by the 2025 workshop at MBZUAI
"Rethinking the Role of Bayesianism in the Age of Modern AI"
NotebookLM

Predictive Execution

LLMs excel at complex reasoning and predictive token generation. But token-level confidence does not equal task-level certainty.



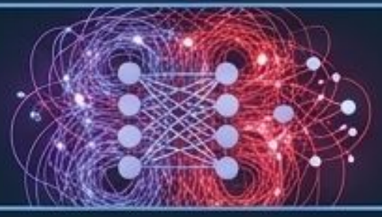
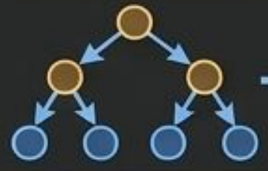
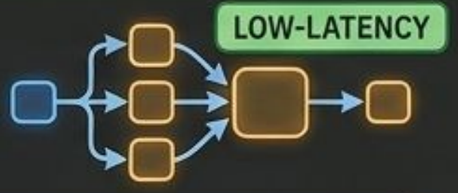
Decision Under Uncertainty



When an agent acts under uncertainty with contextual costs, it requires a decision-relevant latent state to account for shifts in the user, task, or tool ecosystem.

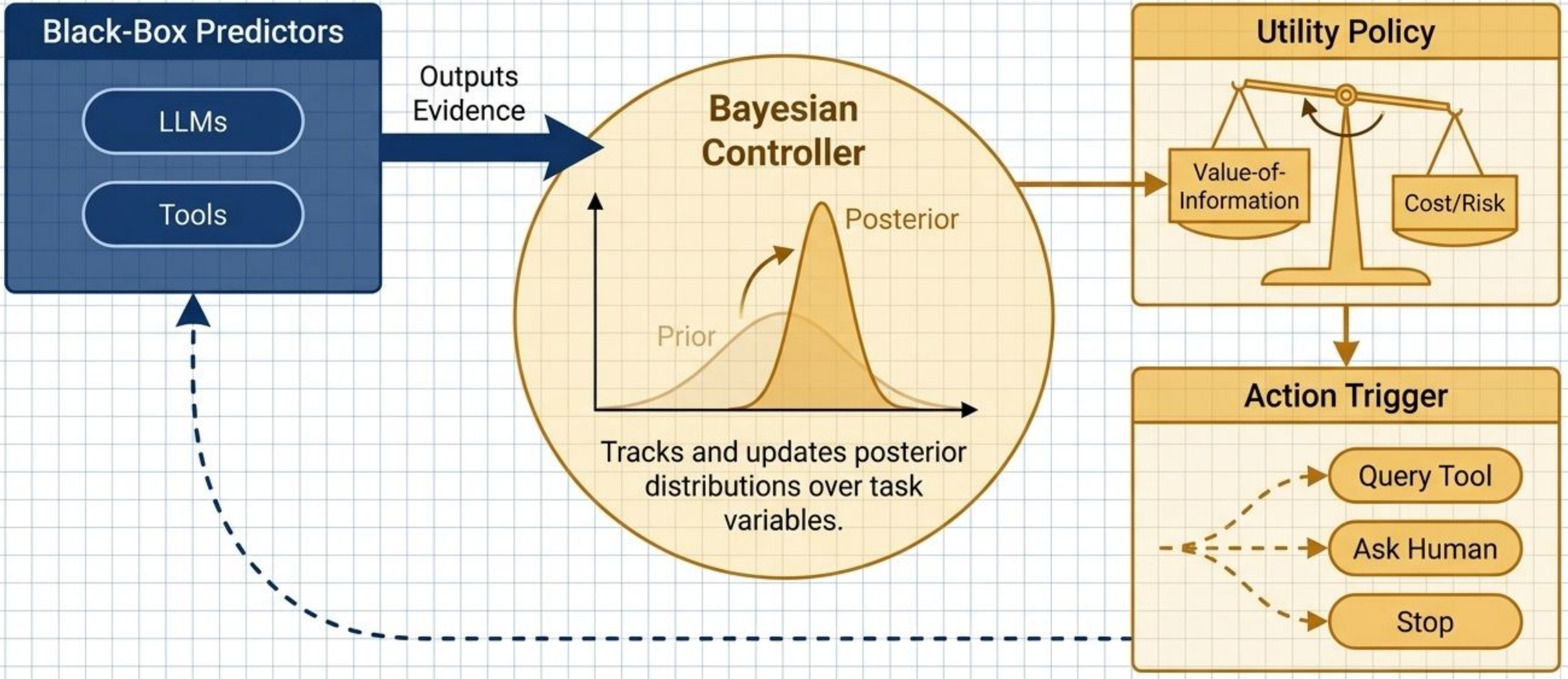
Prediction is about generating the next plausible step.
Orchestration is about routing, stopping, escalating, and allocating budget under risk.

The control layer is where Bayesian principles belong.

	⊗ ~~BAYESIAN LLMs~~	✓ BAYESIAN ORCHESTRATION
Target of Update	Internal LLM Parameters 	**Control Layer Latent Variables** 
Uncertainty Handled	Syntactic (Token-level)  Token distribution	**Semantic** (Task/Decision-level)  Task outcome
Role of the LLM	The solitary decision-maker  Uncertain command	**A predictive evidence generator** 
Computational Overhead	Prohibitive & Unbounded 	**Feasible & Low-Latency** 

Takeaway: Treat LLMs as black-box predictive components. Their outputs are probabilistic evidence fed into an **independent** Bayesian controller.

The **Bayesian Control Loop** separates evidence from decision making.



Triggers tools only when expected improvement outweighs the cost.

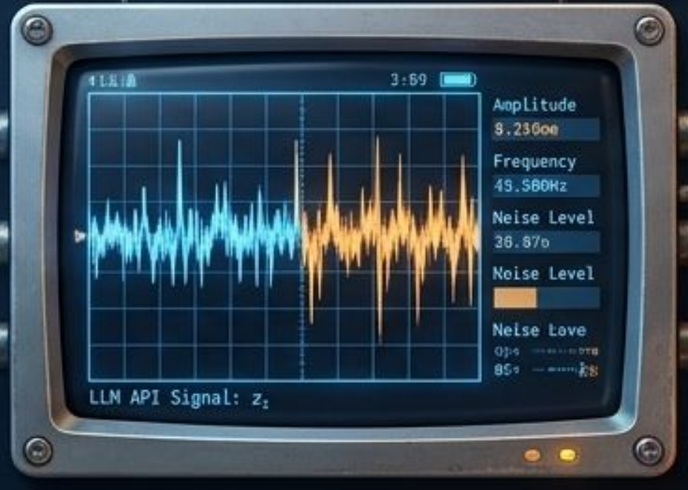
Every agent output acts as a noisy sensor reading.

$$r_t(y) \propto r_{t-1}(y) \cdot p_{i_t}(z_t \mid y)^{\alpha_{i_t}}$$

$r_{t-1}(y)$: The Prior Belief

The orchestrator's confidence in the task outcome before querying the agent.

$p_{i_t}(z_t \mid y)$: The Evidence



The observation model mapping the raw agent message (z_t) to a likelihood of success.

α_{i_t} : The Reliability Filter

A crucial source-specific weight. Down-weights evidence from weaker or biased sources so the system does not produce overconfident updates.

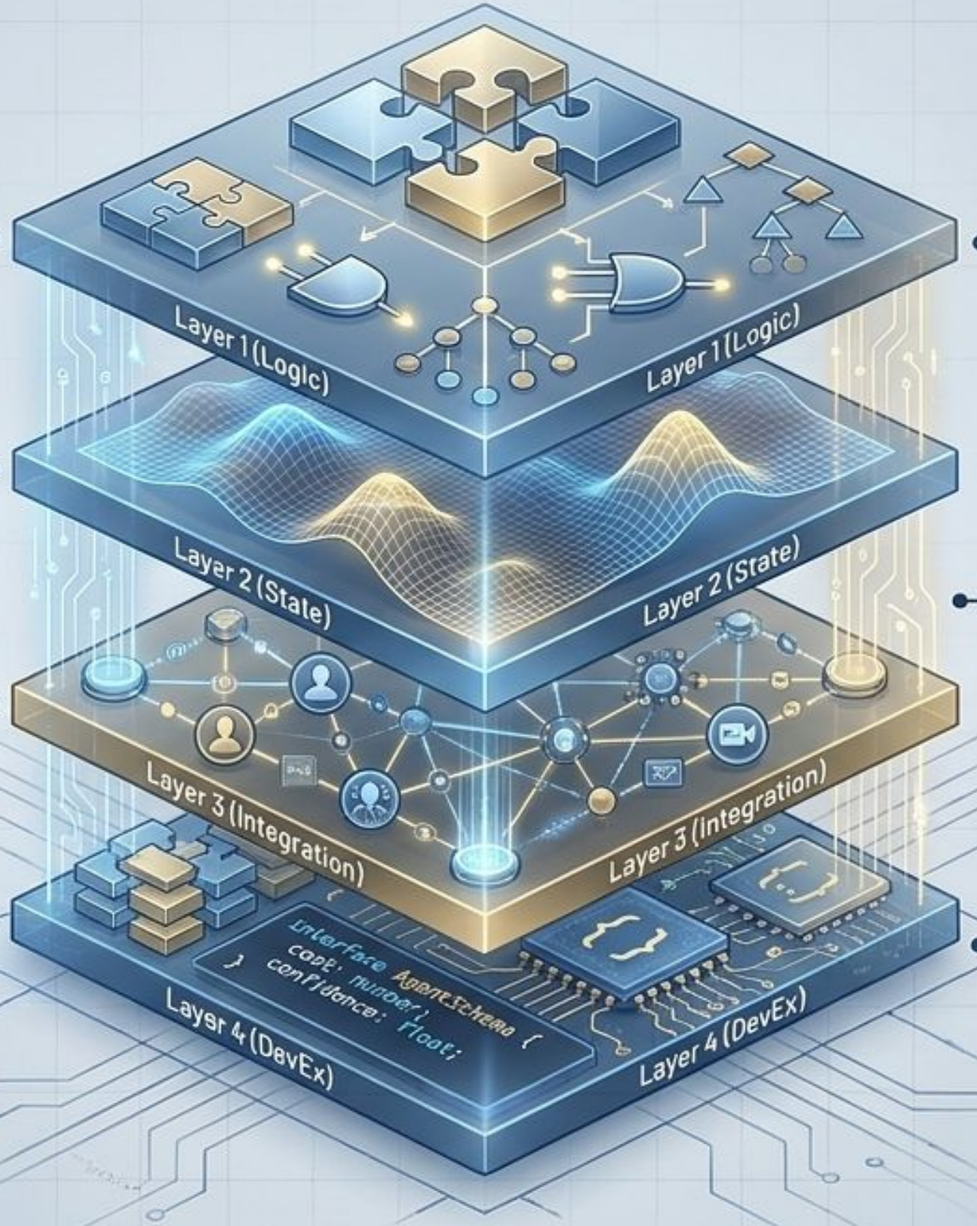
An architecture built for modern human-AI software stacks.

1. Reasoning about utilities and costs:
Modelling privacy risks and call costs as dynamic components.

3. Efficient interaction history integration:
Bayesian distillation replacing massive context windows.

4. Human-AI and multi-agent integration:
Treating human feedback as probabilistic observations.

5. Industry alignment:
Built on typed schemas in TypeScript and Python.

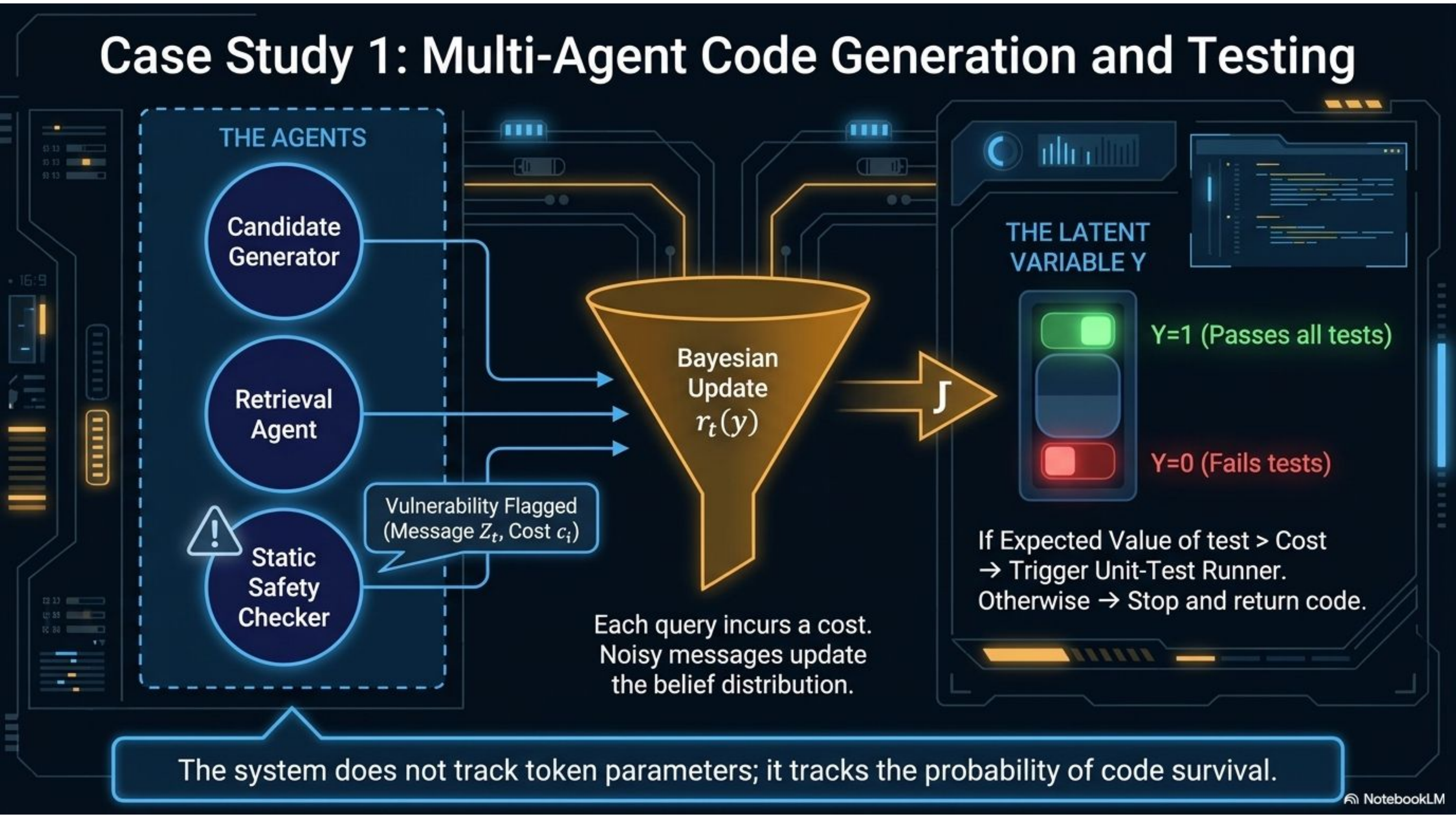


2. Improved decision making with low overhead:
Operating under fixed computational budgets.

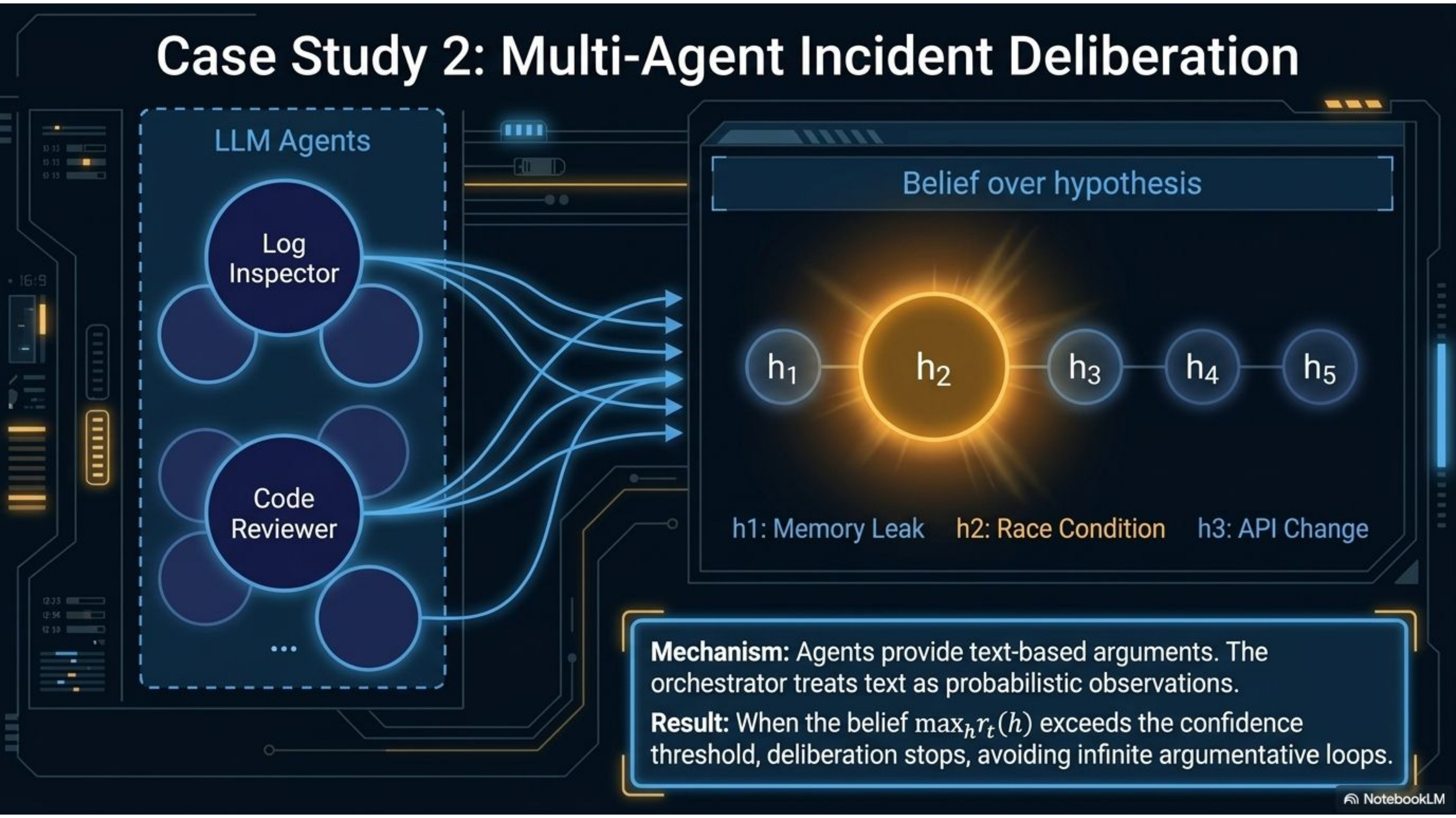
6. Multimodal readiness:
Text, video, and audio evaluated as equal evidence.

7. Accessibility without Bayesian expertise:
Exposing only simple cost and confidence dials to developers.

Case Study 1: Multi-Agent Code Generation and Testing



Case Study 2: Multi-Agent Incident Deliberation



Mechanism: Agents provide text-based arguments. The orchestrator treats text as probabilistic observations.

Result: When the belief $\max_h r_t(h)$ exceeds the confidence threshold, deliberation stops, avoiding infinite argumentative loops.

Case Study 3: Bayesian Routing Across Task Ecosystems

A unified theory of operation across any agentic use case.

	Code Generation	Incident Deliberation	Ecosystem Routing
The Latent Variable	Task Outcome ($Y \in \{0,1\}$)	Discrete Hypothesis ($h \in \mathcal{H}$)	Agent Competence (ψ_i)
The Evidence Source	Code Snippets & Test Logs	Conversational Arguments	Historical Success Labels
The Bayesian Action	Stop / Purchase more tests	Escalate to human / Conclude root cause	Select tool (Explore vs Exploit)

Regardless of the domain, the LLMs remain non-Bayesian generators.
The Bayesian structure dictates what to track, what to trust, and when to act.

Inside the Control State Core.

Task Outcomes & Hypotheses

Tracking the probability of ultimate success based on current evidence.

Cost & Risk Profiles

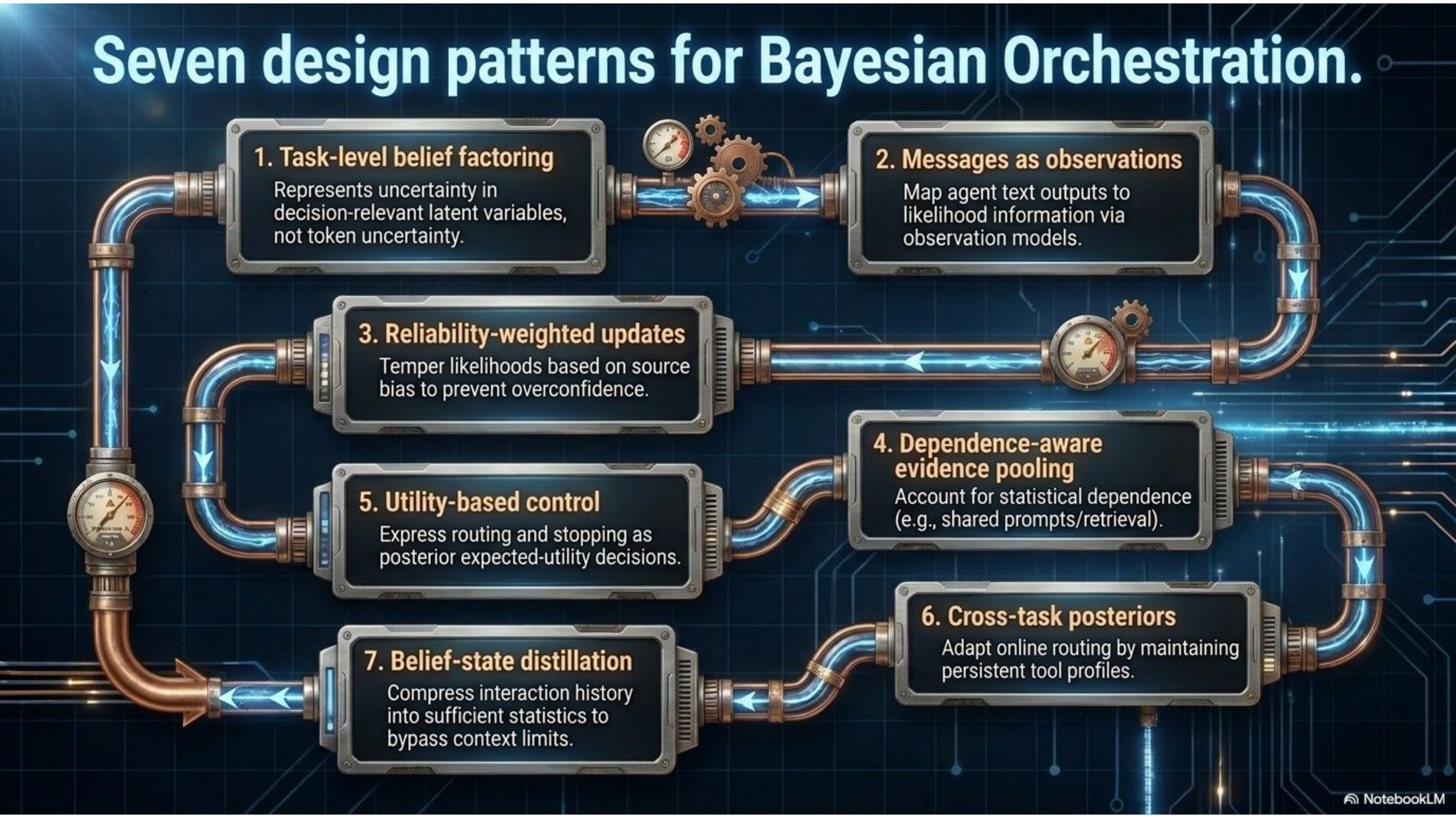
Dynamically updating budget allocation based on the asymmetric costs of errors.

Tool & Agent Reliability

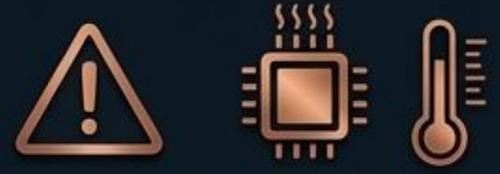
Continuously modelling the trustworthiness of black-box LLMs and external APIs.

CONTROL STATE CORE

Seven design patterns for Bayesian Orchestration.



Known risks and engineering mitigations

RISK	ISSUE	MITIGATION
 High-Dimensional Messages	Observation models mapping text to likelihoods can be misspecified	Continuous recalibration from measurable outcomes and likelihood tempering
 Correlated Evidence	Repeated tool calls to the same LLM yield dependent evidence, creating false confidence	Dependence-aware evidence pooling and conservative updating
 Computational Choke Points	Exact "Value-of-Information" calculations require expensive multi-step planning	Implement one-step approximations or learned surrogates for expected loss reduction

Takeaway: Bayesian control is resilient when updates are conservative and tied to verifiable outcomes

Call to Action

01. Benchmarking

Move beyond sequence prediction.
Implement outcome-based evaluation tracking predictive calibration and Value-of-Information (Vol) metrics.

02. Control layer modelling

Formalise belief states.
Learn observation models from outcome-labelled logs and recalibrate online.

Bayes-Consistent AI Systems

03. Engineering & deployment

Expose simple user controls (confidence/cost dials).
Distil interaction history to manage context windows.

04. Theory for orchestration

$$G(\mathbf{x}) = \prod_{i}^{\infty} \frac{P(x_i|\mathbf{x})}{\partial x_n}$$

Develop decision-focused generalisation models.
Use Bayesian bandits for routing under non-stationarity.

The question is not whether an LLM ‘is Bayesian’.
The question is whether your AI system behaves as a **coherent decision maker** under uncertainty.

Let LLMs prioritise predictive capability. Let the orchestration layer prioritise utility, cost, and truth.