

From Protocol to Practice: Nodes, Not Agents in Human-in-the-Loop LLM Swarms

The MSCFT Ecosystem and its No-Code Templates (MSCFT, SENTINEL, FORGE,
MATHEMATICS) for Forecasting, Coding, Security, and Mathematics

Brian Helip

Santa Monica College

Phi Theta Kappa

August 25, 2025

Table of Contents

Abstract	3
Chapter 1. Introduction	3
Chapter 2. Background and Framework	4
Chapter 3. Technical Implementation	6
Chapter 4. Case Studies and Examples	10
Chapter 5. Discussion	12
Chapter 6. Conclusion	15
References	16

This paper argues that large language models must be treated as tools, not oracles, and that their reliability depends on structured, human-in-the-loop protocols. Building on the MSCFT framework introduced in earlier work, it presents four custom GPT templates—MSCFT, SENTINEL, FORGE, and MATHEMATICS—as proof of concept for a node-based swarm architecture. Through practical testing and analysis, the paper demonstrates that template-driven nodes can be secure, scalable, and consensus forming, offering a path forward for multi-model alignment that emphasizes structure, reproducibility, and oversight.

Chapter 1. Introduction

Large language models (LLMs) have rapidly evolved from experimental systems to widely deployed tools across research, business, and security environments. Their capabilities in reasoning, problem-solving, and pattern recognition are often described in near-autonomous terms, with industry narratives emphasizing “AI agents” that can independently plan, act, and deliver solutions. Yet this framing obscures a fundamental reality: LLMs are not oracles. They are tools. As with any tool, their value depends on structured use, human oversight, and an architecture that acknowledges both their strengths and their limitations. Just as a scalpel in the hands of an untrained physician can cause harm rather than healing, so too can an unstructured LLM produce unreliable or misleading outcomes if not guided by expertise and discipline.

This paper advances that principle by introducing and evaluating four human-in-the-loop templates that extend and operate the MSCFT protocol described in my earlier work, *From coordination failure to scalable AI swarms: the MSCFT protocol for structured forecasting and multi-agent alignment among large language model systems. Whereas the first paper focused on protocol theory—how a swarm of LLMs could be coordinated through structure and consensus, this second paper turns to practice. It documents how templates were designed, implemented as custom GPTs, and tested in live environments.

The four templates- MSCFT, SENTINEL, FORGE, and MATHEMATICS, are each specialized for a domain of structured reasoning. MSCFT governs forecasting and swarm consensus; SENTINEL supports structured threat and security analysis; FORGE enables code development

and refactoring; and MATHEMATICS supports formal problem-solving in quantitative domains. Each is built on the same guiding principle: an LLM should never operate without human validation, and its output must be constrained by structured input. Collectively, these templates can be understood as specialized instruments in a toolbox: one built for forecasting, one for security review, one for coding, and one for mathematical reasoning. Alone, each tool is valuable; together, they form a coherent set capable of addressing diverse challenges.

By building and testing these templates, this paper demonstrates how LLMs can be transformed from unstructured text predictors into disciplined instruments within a multi-node swarm. In doing so, it shows how the theoretical framework of Paper 1 can be made operational, scalable, and secure.

Chapter 2. Background and Framework

This paper builds directly on my earlier work, **From coordination failure to scalable AI swarms: the MSCFT protocol for structured forecasting and multi-agent alignment among large language model systems**. In that first paper, I argued that forecasting with large language models requires more than aggregation; it requires a structured protocol that coordinates multiple model instances, guides them toward consensus, and maintains human oversight. That work introduced a node-oriented architecture in which distinct reasoning roles were aligned through swarm consensus to generate disciplined foresight.

The immediate impetus for this second paper came from a series of Zoom meetings with CCCM. During these discussions, several professors and researchers in computer science expressed the view that LLM swarms could not achieve consensus at a specific point. That skepticism became the challenge that shaped the next phase of my work: identifying the conditions under which multiple model instances could, in fact, come to stable agreement, and designing the protocol structures needed to make that agreement both reliable and reproducible.

The conceptual turning point came when I encountered OpenAI's SWARM protocol, which described orchestrating multiple specialized models ("agents") toward collaborative tasks. Although the terminology emphasized agents, the underlying mechanics align directly with what

I define here as nodes. This reinforced my shift to a node-based vocabulary and showed how structured orchestration could bridge theory and practice.

This work also emerges from a longer lineage of structured forecasting and collective intelligence research. At HRL Laboratories in Malibu, the BARD via Delphi system and the SWARM Project pioneered structured elicitation and Bayesian reasoning for group intelligence. Under IARPA, projects such as CREATE and FOCUS extended these ideas, exploring how disciplined methods could improve the accuracy of distributed forecasting groups. More recently, Good Judgment Open's Project 2.0 for COVID-19 forecasting provided a live testbed for structured protocols in a real-world environment of uncertainty and urgency. Together, these projects established much of the intellectual foundation that MSCFT formalizes into a no-code, template-driven system.

Further clarity was provided by recent research that analyzed the behavior of large language model societies. This work demonstrated that multiple LLMs can reach consensus, but only under bounded conditions, with group stability dependent on model capability and group size. This reinforced the premise that consensus is not impossible, as skeptics argued, but that it requires structure, human oversight, and careful attention to system limits. These insights align directly with the MSCFT protocol's emphasis on bounded roles, protocol-guided interaction, and reaffirmation checkpoints.

Another critical refinement introduced in this paper is terminological. Popular AI discourse has embraced the word "agent" to describe instantiated models, but this language is misleading. An "agent" suggests autonomy and independence, when in practice large language models operate as coordinated, constrained participants within a swarm. The more accurate term is node. A node is a bounded unit that executes a defined role, compares its outputs with other nodes, contributes to consensus, and depends on both protocol and human validation. Within this paper, the term node is adopted exclusively.

When this paper uses the term SWARM, it follows the definition emerging in recent LLM research, where multiple models interact as a collective system. In these systems, stability depends on structured protocols that govern how nodes exchange information and converge toward consensus. Without such structure, group size can exceed critical thresholds, leading to instability and epistemic drift. This usage of swarm is therefore both technical and precise: it

highlights the difference between autonomous “agents” and coordinated “nodes” operating under bounded conditions.

Four specialized templates embody this node framework: MSCFT for forecasting and consensus, SENTINEL for security and threat review, FORGE for code development and refactoring, and MATHEMATICS for structured problem solving. Each template defines a clear role and enforces a human-in-the-loop process. Individually, these templates demonstrate the value of structured constraints; collectively, they demonstrate how a swarm of nodes can be coordinated into a system that produces reliable outcomes.

This background establishes the intellectual continuity between the projects that informed MSCFT, the first paper that defined it, and the present work that operationalizes it. It also clarifies the need for terminological precision: nodes, not agents; consensus, not autonomy; and structure, not unbounded generation.

Chapter 3. Technical Implementation

The MSCFT Ecosystem and its No-Code Templates (MSCFT, SENTINEL, FORGE, MATHEMATICS) for Forecasting, Coding, Security, and Mathematics

The MSCFT protocol and its companion templates were designed as no-code systems, allowing them to be deployed without reliance on external libraries, third-party updates, or specialized software installations. This design choice was intentional: it minimizes attack surfaces in high-security contexts and ensures that the protocol can be applied in any environment where text input and output are possible. By treating templates as structured instructions rather than software packages, the MSCFT ecosystem provides a lightweight but robust architecture for aligning large language models.

Definition: Node (for swarm-based LLM systems)

A node is a structured instantiation of an LLM operating under explicit constraints within a swarm. Properties: (1) task-specific role (e.g., forecasting, coding, math, security); (2) protocol-guided by templates (e.g., MSCFT, SENTINEL, FORGE, MATHEMATICS); (3) consensus-oriented—compares outputs with other nodes and aligns; (4) nested potential—may

contain internal sub-nodes (e.g., A–Z within FORGE); (5) human-in-the-loop—always subject to human validation. In this paper, “node” is used instead of “agent” to emphasize coordination under constraint rather than autonomy.

The overall structure of the swarm, including the four primary custom GPT nodes and the nested A–Z sub-nodes within FORGE, is illustrated in Figure 1.

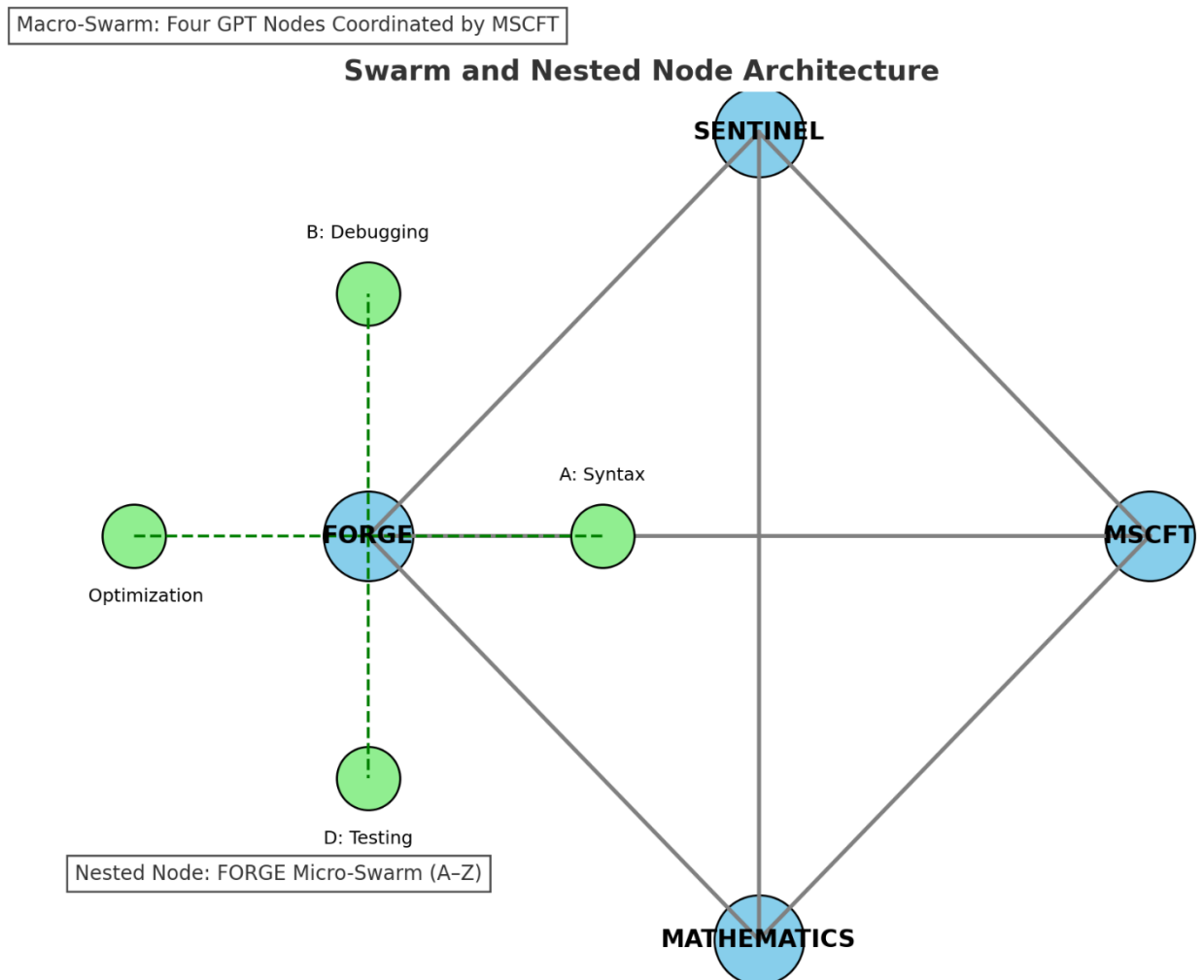


Figure 1

Nested node architecture showing the swarm-level design (MSCFT, SENTINEL, FORGE, MATHEMATICS) and the A–Z sub-nodes inside FORGE.

- MSCFT: governs structured forecasting and swarm consensus.
- SENTINEL: supports structured review of security and adversarial threats.
- FORGE: enables code generation, refactoring, and computational testing.
- MATHEMATICS: guides structured problem solving in formal and quantitative domains.

A custom GPT is built by embedding the template directly into the system and behavior settings, constraining how the model interprets inputs and delivers outputs. This process requires no code and no external dependencies, only a careful mapping of the template instructions into the model's configuration. Once established, each node can operate independently or in coordination with other nodes under the MSCFT protocol.

One innovation demonstrated through this work is the principle of "nested node architecture". At the swarm level, each template-driven GPT is a node. Within a node, however, further subdivision is possible. For example, the FORGE GPT can be organized internally into sub-nodes labeled A through Z, each with a distinct role such as syntax validation, debugging, optimization, or testing. This nested structure permits hierarchical specialization: a node can itself contain a micro-swarm, enabling both inter-node consensus across the swarm and intra-node specialization within a single GPT.

Practical testing has validated the viability of this design. FORGE was deployed in Google Colab notebooks to manage Monte Carlo simulations and refactoring tasks. It successfully executed high-level coding operations, demonstrating the reliability of the template structure. Limitations encountered were not due to the node itself but to external access barriers, such as the inability to obtain API or CRN keys for testing with IBM's quantum computing platform. This distinction highlights an important point: the architecture proved functional, but infrastructure constraints outside the node's design sometimes limited execution.

The MSCFT repository on GitHub now documents all four templates, providing a reference point for replication and adaptation. The architecture is explicitly extensible: although four templates have been tested, the system can scale up to 26 nodes using the alphabetic schema as a baseline. The use of no-code text templates means that new nodes can be created and integrated without risk of breaking code dependencies or introducing vulnerabilities. This extensibility

positions MSCFT as a secure, adaptable framework capable of supporting diverse applications across forecasting, security, coding, and mathematics.

The technical implementation described here demonstrates that structured, no-code templates can be transformed into custom GPT nodes that are both reliable and scalable. The nested node model provides a path for expansion, while the swarm-level architecture ensures coordination and consensus. This balance of modularity, security, and extensibility forms the operational backbone of the MSCFT ecosystem.

For reference, the four custom GPTs built from these templates are summarized below, with share links provided for direct access and replication.

MSCFT (Forecasting) – A structured forecasting custom GPT that improves prediction accuracy by standardizing context, constraints, and output format. Ideal for consensus forecasting, competitive platforms, and decision-support scenarios.

[<https://chatgpt.com/g/g-689eba419048819197095cba703a6a1d-mscft-v4-2>]

FORGE (Coding) – A language-agnostic code generation custom GPT that reduces hallucinations, enforces consistent logic, and supports multiple languages including Python, C/C++, Java, and JavaScript/TypeScript.

[<https://chatgpt.com/g/g-689ea1c3aa6c81919683d75103b09bc3-forge-v-1-0>]

MATHEMATICS – A calculation and formula custom GPT providing immediate access to structured geometry, algebra, and applied math operations for technical, engineering, and scientific use cases.

[<https://chatgpt.com/g/g-689f321989848191b020bb58a84521ab-mscft-mathematics-solver>]

SENTINEL (Security) – A structured security evaluation custom GPT for assessing risks, vulnerabilities, and adversarial scenarios, adaptable for both defensive and investigative applications.

[<https://chatgpt.com/g/g-689f39097bbc819194c2a9b967959a44-sentinel-v1-0>]

These custom GPTs derive from the MSCFT protocol and its companion templates, ensuring a consistent underlying design while allowing domain specific specialization in forecasting, coding, mathematics, and security.

Chapter 4. Case Studies and Examples

The strength of a protocol is measured not only by its theoretical coherence but also by its practical application. To evaluate the performance of the four custom GPT nodes, MSCFT, SENTINEL, FORGE, and MATHEMATICS, each was tested in live environments and real tasks. These case studies demonstrate both the utility of the nodes and the boundaries of their effectiveness.

4.1. MSCFT in Forecasting and Alignment

The MSCFT node was deployed in active forecasting contexts, including health and technology challenges such as the MedHELM leaderboard competition and the Roche AI Challenge. It was also tested in controlled demonstrations, such as a Grok 3 alignment test, where MSCFT prevented harmful output and ensured safe predictions when given structured input. Together, these cases confirm MSCFT's ability to function as an alignment framework and behavioral governor, showing how structured forecasting can reduce individual error while highlighting the limits of consensus when group size or capability thresholds are exceeded.

4.2. SENTINEL for Security Threat Review

The SENTINEL node was designed for structured evaluation of adversarial risks. In testing, it was used to review potential vulnerabilities in multi-model environments, emphasizing its no-code design as a security safeguard. SENTINEL demonstrated that disciplined prompts can guide LLMs toward rigorous threat assessments without introducing additional attack surfaces. Its structured evaluation sequence has been applied in cyber, physical, and geopolitical security scenarios, underscoring its adaptability to high-security contexts.

4.3. FORGE in Computational Testing

The FORGE node was tested in Google Colab notebooks on computational tasks such as Monte Carlo simulations, code refactoring, and controlled experiments with neural architectures. It successfully executed high-level coding, verifying that template-guided GPTs can function as reliable development assistants. Attempts to extend testing to quantum computing environments were constrained by external factors, such as the inability to secure API or CRN keys from IBM. This distinction highlights a key lesson: infrastructure bottlenecks, not protocol design, often limit performance in practice.

4.4. MATHEMATICS in Quantitative Reasoning

The MATHEMATICS node was used to structure problem-solving tasks in algebra, statistics, and visualization. Practical use cases included generating both simple plots and more complex gene-style visualizations with ggplot2, as well as supporting CNN-based modeling. Beyond experimental runs, MATHEMATICS has also been applied in clinical diagnostics and scientific research, where structured reasoning reduced epistemic risk, improved reproducibility, and integrated well with existing technical workflows. These examples validated the ability of a single node to enforce mathematical discipline and prevent drift in outputs.

4.5. MSCFT in Infrastructure Modernization

MSCFT has also been applied to real-world infrastructure challenges, including a use case for modernizing the U.S. air traffic control system. By leveraging its node-based design, audit traceability, and fault isolation capabilities, MSCFT demonstrated potential for enhancing national-scale systems where reliability and accountability are critical. This case showed that the protocol extends beyond forecasting and analysis into operational domains that demand structured coordination and validation.

4.6. Lessons from Case Studies

Across all case studies, several lessons emerged. First, template-driven nodes reliably constrained LLM behavior in ways that improved interpretability and reduced unstructured drift. Second, the swarm model allowed cross-checking of outputs, reinforcing accuracy when nodes operated under human-in-the-loop supervision. Third, external constraints—such as infrastructure access, scoring practices, or institutional bottlenecks—proved as influential as model performance, underscoring the importance of context in applied forecasting and analysis.

Together, these examples show how the MSCFT ecosystem operates not only in theory but in practice, where nodes and their templates generate structured, testable, and reproducible outputs across domains as diverse as healthcare, physics, infrastructure, security, and competitive forecasting.

Additional case study files and demonstration examples, including forecasting exercises, security reviews, computational tests, and infrastructure applications, are available in the MSCFT GitHub repository. Readers can access the full set of examples here:

[<https://github.com/captbullett65/MSCTF/tree/main/examples>]

Chapter 5. Discussion

The case studies described above illustrate both the strengths and limitations of template-driven swarm architectures. Together they reinforce three central claims: that LLMs must be understood as tools rather than oracles, that reliable consensus requires human-in-the-loop structure, and that the correct conceptual vocabulary is nodes, not agents.

5.1. LLMs as Tools, Not Oracles

The most important lesson is that large language models are not autonomous decision-makers. Their value emerges only when they are used as structured instruments under human oversight. The MSCFT, SENTINEL, FORGE, and MATHEMATICS, custom GPTs each demonstrated that without constraint, outputs drift toward error or inconsistency. With structured templates in place, however, the reasoning process becomes bounded, testable, and reproducible. Like a scalpel in the hands of a trained surgeon, the utility of an LLM depends entirely on how skillfully it is guided.

5.2. Human-in-the-Loop Design

Every custom GPT in the MSCFT ecosystem is explicitly human-in-the-loop. This design acknowledges a critical truth: mistakes will always occur. No model, regardless of scale, can be assumed to deliver perfect answers. The human role provides the validation layer that ensures

output remains actionable. This safeguard echoes earlier lessons from forecasting research, where disciplined human judgment was needed to interpret probabilistic results.

5.3. Nodes, Not Agents

The misuse of the word “agent” remains one of the clearest examples of how hype distorts understanding. In the context of LLM SWARMs, the correct unit is the node: a bounded, protocol-guided participant in a swarm. Agents imply autonomy and independence, while nodes emphasize coordination, specialization, and validation. The MSCFT architecture extends this principle with nested nodes, demonstrating that a single node, such as FORGE, can itself host a micro-swarm of sub-nodes. This vocabulary shift is foundational. It reframes how practitioners think about design, moving away from marketing-driven anthropomorphism and toward technically precise swarm consensus.

Even OpenAI’s early documentation of SWARM framed each participant as an agent, but the architecture itself demonstrates role-bounded participants—what this paper calls nodes. The language of agents risks overstating autonomy, while nodes highlight the collaborative, constrained reality.

It is important to emphasize that SWARM here is not a metaphor but a reference to a specific research tradition. Recent studies have shown that LLM collectives can achieve coordination only under structured conditions, and that stability collapses if group size or role definition is left unmanaged. MSCFT addresses this limitation directly by bounding each node’s role and aligning outputs through structured consensus. In doing so, it demonstrates how SWARMs can be stabilized in practice, not just described in theory.

5.4. Accuracy vs. Scoring

Testing also revealed the distinction between true accuracy and the scoring metrics used to evaluate it. In competitive or platform-based settings, scoring often rewards alignment with crowd medians or probabilistic averages. While useful for standardization, these measures can obscure deeper aspects of foresight. A forecaster who identifies an emerging signal early may appear “inaccurate” if their probability diverges from the median, even if later proven correct. Conversely, adherence to the median may protect scores without advancing insight. Structured

protocols such as MSCFT emphasize that accuracy is not reducible to a single score. Instead, it should be understood as disciplined reasoning, reproducibility, and resilience to error over time.

5.5. Security and No-Code Design

The no-code architecture of the MSCFT ecosystem has significant implications for security. By avoiding dependencies on external libraries or constant updates, the system minimizes exposure to attack surfaces and supply-chain vulnerabilities. SENTINEL was designed under precisely these constraints, confirming that structured reviews can be performed in high-security contexts where code execution is not permitted. No-code design is not just convenient — in these environments, it is mandatory.

5.6. Scalability and Infrastructure Constraints

The MSCFT system is inherently scalable. It can expand from four nodes to 26 or more, with nested sub-nodes enabling hierarchical specialization. Yet testing also showed the limits imposed by infrastructure. FORGE’s experiments with quantum computing, for example, were blocked not by design but by access barriers such as unavailable API keys. This distinction is critical: the architecture is functional, but implementation is often constrained by institutional or infrastructural bottlenecks. Future research must address these external limits, whether through open standards, partnerships, or expanded compute access.

5.7. Implications for Multi-Model Systems

Taken together, these lessons suggest a reframing of how multi-model LLM systems should be understood. The future does not lie in ever-larger autonomous “agents” but in structured swarms of specialized nodes coordinated under human oversight. This approach preserves accuracy, enhances security, and ensures scalability. More importantly, it continues the lineage of structured forecasting research — from Delphi and SWARM at HRL to IARPA’s CREATE and FOCUS — by showing that collective intelligence is achieved through structure, not autonomy.

Chapter 6. Conclusion

The work presented in this paper demonstrates how structured, no-code templates can transform large language models from unstructured text predictors into disciplined, human-in-the-loop tools. Building on the theoretical foundation laid out in the MSCFT protocol, four specialized templates—MSCFT, SENTINEL, FORGE, and MATHEMATICS—were implemented as custom GPT nodes and tested in practical environments. Together, they form a proof of concept for a modular ecosystem in which each node carries a defined role, can scale through nested architectures, and participates in swarm consensus without reliance on external code dependencies.

Several conclusions follow from this research. First, LLMs achieve their greatest reliability when guided by structured protocols and validated through human oversight. Second, the correct conceptual vocabulary for these systems is nodes, not agents, as their value derives from coordination under constraint rather than autonomy. Third, template-driven nodes are secure and adaptable, precisely because they are no-code designs that avoid the vulnerabilities of library-based systems. Finally, the scalability of this architecture, from four nodes to 26 or more, demonstrates its potential as a flexible framework for forecasting, coding, security evaluation, and quantitative reasoning.

The broader implication is that the future of multi-model systems lies not in marketing promises of autonomous agents but in the careful design of structured swarms. By combining discipline, modularity, and oversight, these swarms can achieve consensus, preserve accuracy, and adapt to high-security contexts. In doing so, they extend the lineage of structured forecasting research while addressing the challenges unique to today's large-scale models.

References

- Ahmed Tokyo. (2025). OpenAI's SWARM protocol explained.
https://ahmedtokyo.com/blog/2025/ai_openai_swarm
- Anthropic. (2025). *Grok 3 evaluation and alignment testing* [Public demonstration material].
- De Marzo, A., Castellano, G., & García, E. (2025). Agent societies and epistemic instability in LLM swarms (arXiv:2409.02822v4). *arXiv*. <https://arxiv.org/abs/2409.02822>
- Federal Aviation Administration (FAA). (2025). *Modernizing U.S. air traffic control systems: Application of AI frameworks*. <https://www.faa.gov>
- Google Colab. (2025). *Colaboratory: Machine learning and computational notebooks*.
<https://colab.research.google.com>
- Helip, B. (2025). *From coordination failure to scalable AI swarms: The MSCFT protocol for structured forecasting and multi-agent alignment among large language model systems* [Unpublished manuscript].
- Helip, B. (2025). *Master SWARM Consensus Forecasting Template (MSCFT)* [GitHub repository]. <https://github.com/captbullett65/MSCFT>
- Helip, B. (2025). *MSCFT: Custom GPT implementations (MSCFT, SENTINEL, FORGE, MATHEMATICS)*. <https://chatgpt.com>
- HRL Laboratories. (2018). *SWARM Project and BARD via Delphi* [Research reports].
 Malibu, CA: HRL Labs.
- IBM Quantum. (2025). *Qiskit Runtime and API access*. <https://www.ibm.com/quantum>

IARPA. (2016). *CREATE Program: Crowdsourcing Evidence, Argumentation, Thinking and Evaluation*. <https://www.iarpa.gov/research-programs/create>

IARPA. (2017). *FOCUS Program: Forecasting Counterfactuals in Uncontrolled Settings*. <https://www.iarpa.gov/research-programs/focus>

OpenAI. (2025). *SWARM (experimental, educational)* [GitHub repository]. <https://github.com/openai/swarm>

Roche. (2025). *Roche AI Challenge*. <https://www.roche.com>

Stanford Center for Research on Foundation Models (CRFM). (2025). *MedHELM leaderboard*. <https://crfm.stanford.edu/medhelm>

Tetlock, P., & Gardner, D. (2015). *Superforecasting: The art and science of prediction*. New York: Crown.