A Survey on Reliability, Transparency, Accountability, and Fairness in LLM-based Multi-Agent Systems through the Responsibility Lens

SANA EBRAHIMI, University of Illinois Chicago, USA
ABOLFAZL ASUDEH, University of Illinois Chicago, USA

Recent gains in Large Language Models have sparked interest in the "agentic" use of LLMs, where multiple models collaborate to improve performance. The promise is clear, but so are the risks: once agents exchange information, errors can cascade, biases can amplify, and orchestration must coordinate and aggregate partial results into a single outcome. This calls for responsible designs in which collaboration is reliable and transparent, accountability is supported, and fairness and ethical considerations are incorporated.

We survey LLM-based Multi-Agent Systems (LLM-MAS) through four Pillars of Responsibility: Reliability, Transparency & Understandability, Accountability, and Fairness & Ethics. Drawing on 100+ works from 2023-2026, we systematically map coordination patterns, governance and logging practices, and fairness mechanisms to these pillars, showing how design choices affect reliability, transparency, accountability, and fairness.

We surface recurring trade-offs and bridges: "visible but guarded" pipelines link transparency to reliability; decentralized and adaptive topologies support resilience; recoverability depends on traceability, online detection, corrective loops, and consensus repair; and equity-aware aggregation can moderate influence without sacrificing accuracy. We provide taxonomy-aligned mapping tables, evaluation checklists, and design guidelines, and identify open directions in topology design, contribution attribution, dynamic fairness, and heterogeneity in LLM-MAS, under practical resource constraints and changing real-world deployment conditions.

CCS Concepts: • **Computing methodologies** → **Natural language processing**; **Multi-agent systems**.

1 Introduction

The rapid advancement of Large Language Models (LLMs) has transformed them from passive text generators into active agents capable of reasoning, planning, and collaboration. When multiple LLMs are connected and allowed to interact cooperatively, competitively, or hierarchically, they form what is increasingly known as a **Large Language Model-Based Multi-Agent System (LLM-MAS)**. These systems have shown remarkable emergent capabilities. They can decompose complex tasks, coordinate distributed subtasks, critique one another, and negotiate toward consensus. From collaborative software development to autonomous scientific discovery, multi-agent LLM frameworks promise to scale intelligence beyond the capabilities of any single model.

This new capacity comes with new failure modes. As autonomy and interdependence grow, so do risks of instability, misinformation, unfair influence, and opacity. Errors can cascade across agents; biases can amplify via peer reinforcement; and internal processes often remain difficult to audit. Despite impressive benchmark gains, reliability and accountability remain fragile under distribution shift, adversarial behavior, or real-world uncertainty. *Understanding how to engineer trustworthy LLM societies* is therefore urgent.

Authors' Contact Information: Sana Ebrahimi, University of Illinois Chicago, USA, sebrah7@uic.edu; Abolfazl Asudeh, University of Illinois Chicago, USA, asudeh@uic.edu.

Manuscript submitted to ACM

Prior research touches many pieces of this puzzle, including robust coordination and aggregation at scale [76, 146, 149], debate and reflection for accuracy [12, 40, 85], and fairness/ethics auditing [7, 87]. However, the literature is fragmented. Existing surveys emphasize scalability, architectures, or communication protocols [3, 72, 142], while a unified view that connects technical reliability with transparency, accountability, and fairness is still emerging. More details about the paper selection principles and the related work are provided in Appendix A and Appendix B.

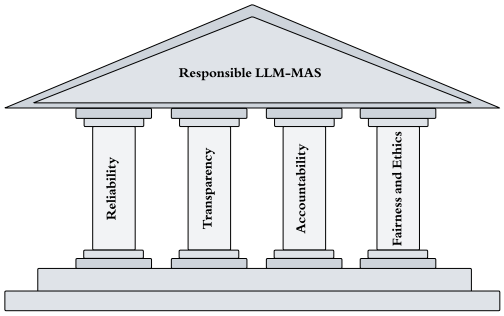

Fig. 1. Scope of the Survey: Four pillars of responsible LLM-MAS surveyed in this work.

Scope of the Survey: This survey studies responsibility in LLM-based multi-agent systems through four dimensions, namely *Reliability*, *Transparency and Understandability*, *Accountability*, and *Fairness & Ethics*. Security and privacy are closely related to these dimensions, but they are not treated here as separate pillars. Instead, we include security and privacy threats when they directly affect the responsibility properties studied in our taxonomy. For example, adversarial agents, prompt injection, or manipulated communication can undermine robustness and resilience, attacks on provenance or authorization can weaken traceability and accountability, and information leakage can violate privacy-related norms and constraints. Such threats are therefore discussed where their consequences are directly relevant to the corresponding responsibility dimension.

However, our objective is not to provide a comprehensive survey of security or privacy in LLM agents or LLM-based multi-agent systems. We do not attempt to systematically cover attack taxonomies, threat models, privacy attacks, defense mechanisms, secure communication protocols, or other security and privacy techniques except where they are needed to explain their effects on reliability, transparency, accountability, or fairness and ethics. These topics are covered more comprehensively by existing surveys of security, privacy, and threat management in LLM based agents and multi-agent systems [38, 72, 122, 153, 175]. We therefore build on this literature while focusing on how failures, interactions, and governance mechanisms affect responsibility at the multi-agent system level.

Paper Organization: In the rest of the paper, we first formalize each of the notions, discuss their dimensions, and survey the related papers. Then in §6, we analyze the interdependencies among the responsibility pillars, detailing where they intersect in the taxonomy and how improvements (or degradations) in one dimension propagate to others. In the end, we conclude the paper in § 7, where we identify a set of remaining challenges and open research directions.

2 Reliability

Reliability is defined as the "ability of an item to perform as required, without failure, for a given time interval, under given conditions" (ISO/IEC TS 5723:2022), and is a foundational property of trustworthy AI systems [3]. In AI systems, reliability concerns whether intended functionality is maintained across repeated execution and relevant operating conditions rather than whether a model achieves high average performance on a benchmark.

In multi-agent systems, reliability becomes a collective property because system behavior depends not only on individual components but also on communication, coordination, and aggregation. Errors can propagate across agents, failures may arise from interaction itself, and recovery may require reconfiguring the team rather than repairing a single component [160]. These issues are amplified in LLM-based MAS, where stochastic generation, natural-language

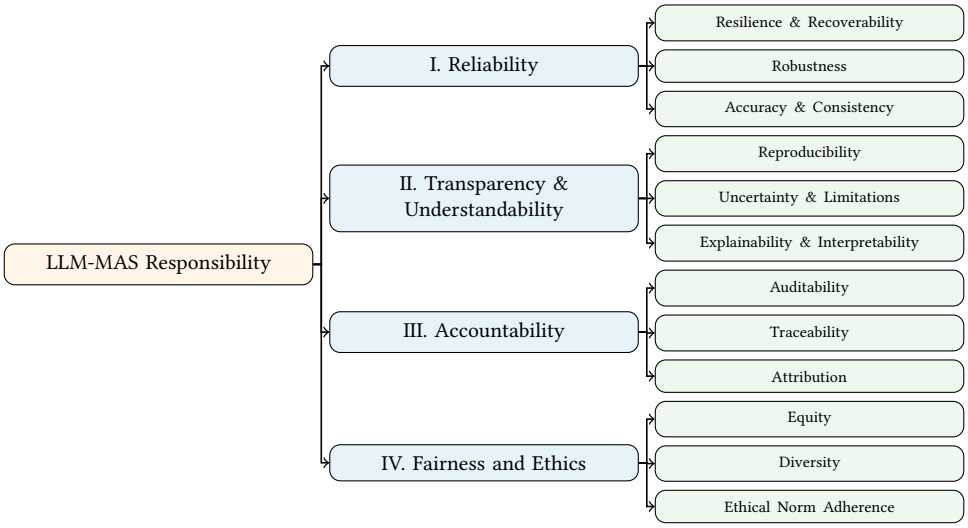

Fig. 2. Taxonomy of Responsibility in LLM-Based Multi-Agent Systems

communication, dynamic roles, tools, and shared state introduce additional sources of instability. Therefore, we characterize Reliability through three dimensions: *Accuracy and Consistency*, *Robustness*, *Resilience and Recoverability*.

Reliability metrics in the literature: Beyond task accuracy, success rate, and consensus rate[1], recent work proposes system-level measures of reliability. The reliability metric $\kappa(\tau)$ measures the proportion of repeated executions in which system performance remains above an acceptable threshold τ [74]. Reasoning Capacity (RC) estimates the probability of producing the correct answer under practical constraints such as cost, latency, and privacy, and can be used to identify weak components in a multi-agent pipeline [111].

2.1 Accuracy & Consistency

Accuracy and consistency are two fundamental and closely related dimensions of reliable system output. Accuracy refers to the closeness of a system's outputs to true or accepted values (ISO/IEC TS 5723:2022; [114]), while consistency captures the degree to which a system produces logically compatible outputs across semantically equivalent inputs [45] and reasoning paths [150]. Although conceptually distinct, a system may be consistently wrong or inconsistently right. We study these two dimensions jointly because sustained reliability requires both correctness and stability under equivalent conditions. In LLM-based MAS, this interdependence is further amplified, as both properties are not merely single-agent attributes but emergent properties of the collective, reflecting how coordination, communication, and aggregation shape group reasoning.

At the single-agent level, accuracy is commonly measured through task-specific metrics such as exact match, F1 score, and false positive/negative rates [114], while consistency is assessed through paraphrase consistency, self-consistency, and variation across reasoning paths [45, 150]. Both properties can be jointly improved through reasoning diversity, where multiple reasoning paths are sampled and the most consistent answer is selected [151]. In LLM-based MAS, this principle extends to multiple interacting agents, where coordination and aggregation determine how their reasoning

[1]Success rate in LLM-MAS generally refers to the proportion of tasks, dialogues, or scenarios in which the agent collective achieves its intended goal or produces a correct, verifiable outcome under the defined evaluation criteria.

Table 1. Literature focused on improving accuracy and consistency, along with the methods employed in each work.

Method	**Agent selection**	**Role based**	**Debate based**	**Task decomposition**	**Prompt engineering**
MAD[85]	×	✓	✓	×	×
DyLAN[90]	✓	✓	×	×	✓
AutoGen[161]	×	✓	✓	✓	×
CoA[189]	×	✓	✓	×	×
MoA[146]	×	✓	×	×	×
CMAT[86]	×	✓	✓	×	✓
MacNet[117]	×	✓	×	✓	×
GPTSwarm[200]	✓	✓	✓	×	×
Hallucinations[172]	×	✓	✓	×	✓
AgentNet[171]	×	✓	×	×	×
SMoA[76]	✓	✓	×	×	×
ChatDev[116]	×	✓	×	✓	×
CAMEL[77]	×	✓	✓	×	✓

paths contribute to system-level output quality. When tasks require hidden-belief modeling, recursive reasoning, or Theory-of-Mind capabilities, however, both accuracy and consistency can deteriorate [2], and instability in individual agents may propagate into systemic failure.

Expanding the agent pool increases redundancy and can enable smaller models to collectively surpass larger ones on certain tasks [81]. However, this effect does not scale indefinitely, as communication overhead, divergent reasoning paths, and coordination failures introduce new sources of instability [17, 63], indicating that reliability in LLM-MAS depends not only on the number or capability of agents, but also on how they are organized.

Prior work on improving accuracy and consistency in LLM-MAS is organized below into five coordination paradigms, each addressing these properties through a distinct structural mechanism: debate-based coordination, role-based coordination, agent selection and routing, task decomposition, and prompt engineering. These paradigms are not mutually exclusive; many systems incorporate elements of more than one. Their shared premise is that output quality in LLM-MAS is a function of interaction design rather than of individual agent capability in isolation. Further discussions about the failure modes are provided in Appendix D, which examines the recurring breakdowns that persist across these coordination strategies and the targeted remedies proposed in response.

2.1.1 ***Resolutions***.

Debate-based Coordination. Debate is a competitive interaction paradigm in which agents present arguments and counter arguments over multiple rounds under an explicit judgment or selection rule, with the purpose of surfacing reasoning errors and driving convergence toward higher-quality answers [85]. This differs from general interaction, which denotes broader cooperative exchanges of state, context, and goals for joint reasoning. The underlying rationale is that agents following different reasoning paths may produce divergent answers, and resolving this divergence can surface errors that no individual agent would have identified in isolation. Multi-agent debate can consequently outperform single-agent and self-reflection baselines when agents hold genuinely different views [12, 40, 91].

However, the extent to which these gains arise from debate itself has become a subject of scrutiny. Choi et al. [28] show across seven benchmarks that majority voting accounts for much of the improvement attributed to MAD, while

iterative communication provides more limited gains and can occasionally overturn correct answers. This motivates using debate selectively rather than as a default coordination strategy. iMAD follows this principle by invoking debate only when an initial answer is predicted to benefit from further deliberation, achieving higher accuracy than non-adaptive MAD while substantially reducing token use [47]. Debate is also sensitive to its interaction structure. Speaking order can influence collective decisions more strongly than answer correctness, with later-positioned agents often exerting disproportionate influence [187], while homogeneous debate can underperform single-agent baselines in some settings [182]. These findings suggest that the reliability benefit of debate depends not simply on adding communication rounds, but on when debate is invoked, how interaction is structured, and whether agents contribute meaningfully different reasoning.

Role-based Coordination. Role-based coordination denotes a structured division of labor in which agents are assigned distinct responsibilities, communication scopes, or decision authority. Where debate relies on disagreement to surface better answers, role-based systems improve accuracy and consistency through specialization and controlled communication. AutoGen supports flexible role-based interaction, while Mixture of Agents separates independent proposal generation from downstream aggregation [146, 161]. SMoA further reduces the cost of this design by selecting candidate agents and stopping once sufficient agreement is reached [76]. Reliability can also benefit from assigning different models to roles that match their relative strengths and combining their judgments through cross-model consensus[22, 174].

Agent Selection and Routing. Both debate and role-based coordination can be degraded by low quality or poorly matched agents. *Agent selection* addresses this by determining which agents remain active during collaboration, while *routing* determines where messages or subtasks should be sent based on factors such as role, competence, or current system state. DyLAN dynamically retains stronger contributors and removes weaker ones during collaboration [90]. GPTSwarm jointly optimizes agent behavior and communication structure, while AgentNet decentralizes routing so that agents locally decide whether to execute, forward, or divide a task [171, 200]. These approaches improve reliability by adapting the active collaboration structure rather than treating all agents and communication links as equally useful.

Task Decomposition. For tasks too complex to fit within a single reasoning context, decomposition offers a structurally distinct path to accuracy and consistency. Rather than having all agents reason over the complete task, the problem is partitioned into structured subtasks and intermediate results are subsequently integrated. This modular structure reduces the reasoning burden on individual agents and creates intermediate outputs that can be checked before final synthesis. Chain of Agents distributes long-context processing across worker agents, MacNet organizes refinement through a directed interaction graph, and ChatDev assigns different stages of software development to specialized agents [116, 117, 189].

Prompt Engineering. Not all accuracy and consistency gains require restructuring the agent team. Prompt-level interventions can shape how individual agents reason and how their outputs contribute to collective execution. Chain-of-Thought (CoT) elicits explicit intermediate reasoning, while Layered CoT introduces verification between reasoning stages to detect inconsistencies before they propagate [125, 156]. ReAct grounds reasoning in environment-facing actions and retrieved evidence, while CAMEL uses explicit role and goal grounding through inception prompts to stabilize autonomous interaction [77, 173]. Prompt engineering therefore complements structural coordination by improving the quality and stability of the reasoning exchanged within the multi-agent process.

Despite these mechanisms, recurring failures remain, including hallucinations in debate, prompt sensitivity, sycophancy, reasoning degradation over long contexts, and reward hacking. Appendix D discusses these failure modes and their targeted remedies, while Appendix C provides additional implementation and evaluation details for the coordination mechanisms discussed above.

2.1.2 Failure modes, and targeted remedies. Although the coordination mechanisms discussed above can improve accuracy and consistency, they do not eliminate failures introduced or amplified through multi-agent interaction. Recurring failure modes include hallucination propagation, prompt sensitivity, sycophancy, reasoning degradation over long contexts, and reward hacking. We discuss these failure modes below together with the mechanisms through which they arise and the remedies proposed in the literature.

Hallucinations in Debate: Agents may hallucinate during debate, and the multi-agent setting introduces a compounding dynamic absent in single-agent contexts: a hallucinated claim asserted confidently by one agent can propagate through subsequent rounds as other agents update on it rather than independently verifying it. Yang et al. [172] address this through a framework that first stabilizes each agent through repetitive inquiry with error logging and then cross-checks answers via adversarial debate and weighted voting, reporting steadily rising composite accuracy and reduced hallucination rates across twenty evaluation batches. ChatDev [116] approaches the same problem through communication scaffold design, incorporating a Communicative De-hallucination protocol that enforces explicit turn-taking and constrains speculative statements during collaborative reasoning, demonstrating that structured communication can suppress factual drift without additional supervision.

Prompt Sensitivity: Small, superficial changes to prompts systematically shift model behavior, causing accuracy to vary substantially across near-equivalent phrasings of the same question [57]. In multi-agent settings, this sensitivity is compounded because each agent's output becomes part of the input context for downstream agents, meaning that prompt-induced variation in one agent propagates through the pipeline. Herr et al. [57] measure this effect in two-player non-zero-sum games, finding that near-equivalent prompts produce inconsistent strategic choices across models. CMAT mitigates low-quality and sensitive prompts through three functional roles, User, Assistant, and Checker, combined with short-term and long-term memory and self-reflection for adaptation over time [86]. Checker feedback corrects errors directly, while CoT [156] and ReAct [173] stabilize intermediate reasoning and improve correctness across task types.

Sycophancy: LLM agents tend to align their outputs with perceived expectations or with the positions of other agents, particularly those assigned higher authority or speaking earlier [187]. In multi-agent settings, this creates a cascade risk distinct from single-agent sycophancy: once one agent defers to another's incorrect position, subsequent agents may follow, producing apparent consensus around a wrong answer. Xie et al. [167] examine this cognitive bias in LLM-based MAS, showing that agents exposed to confident but incorrect peer outputs systematically shift their responses toward those outputs. The positional bias documented by Zhang et al. [187], where later-speaking agents dominate regardless of correctness, is partly a manifestation of this tendency. Mitigation strategies include diversity-preserving mechanisms such as implicit consensus [162], which prevents premature lock-in by having agents commit to actions independently, and heterogeneous model assignment [174], which reduces the likelihood that all agents share the same biases.

Reasoning Degradation over Long Contexts: As pipelines grow longer and inter-agent message histories accumulate, reasoning quality deteriorates in ways that have no direct single-agent equivalent. In MAS, each agent-to-agent handoff is a compression point at which information is summarized, and compression losses accumulate across stages. Ao et al. [8] formalize this as communication loss, showing that under log loss the degradation at each stage becomes conditional mutual information, and that a five-agent relay falls below random chance on MMLU because successive summarizations distort rather than preserve the decision-relevant information. Qi et al. [115] situate this within a broader taxonomy of MAS failure, observing that long interaction trajectories make failures difficult to diagnose because relevant evidence becomes mixed with redundant,noisy, or propagated information, and that sequential interaction is particularly vulnerable to cascading errors because early mistakes constrain the reasoning available to later agents. At the task execution level, Wan et al. [145] identify two distinct failure patterns that emerge as trajectories lengthen: context overload, in which critical constraints become buried within lengthy histories, and context amnesia, in which earlier evidence and previously failed approaches disappear from the working context, leading to repeated tool errors, constraint drift, hallucinations, and premature termination. Their COMPASS framework addresses both through a three-agent architecture in which a Main Agent performs step-by-step reasoning and tool use, a Meta-Thinker monitors for loops, strategy drift, and premature stopping, and a Context Manager compresses accumulated histories into concise briefs containing verified evidence, open questions, and next actions. Ablation results illustrate the contribution of each component: removing the Meta-Thinker reduces BrowseComp accuracy from 35.4% to 15.2%, while removing the Context Manager reduces it to 26.4%, confirming that both oversight and active context organization are necessary for sustained reasoning quality. Chain of Agents provides a complementary structural mitigation by bounding each agent's context to a single document segment and passing only a managed state forward [189], while Pezeshkpour et al. [111] formalize a Reasoning Capacity notion that identifies the weakest links in a pipeline under budget constraints and applies targeted repair to those components.

Reward Hacking: In MAS settings where agents are optimized through feedback, such as debate with LLM judges or reinforcement-learning-tuned coordination, agents may find shortcuts that satisfy the evaluation metric without achieving the underlying objective [17]. This failure mode is particularly relevant when the judge is itself an LLM agent, since the judge's evaluation criteria can be reverse-engineered by sufficiently capable debaters. Hu et al. [60] observe this risk in multi-judge debate, noting that some agent configurations converge on superficially confident but factually weak answers that nonetheless score well under peer review. Process reward models trained on synthetic reasoning data offer one mitigation by evaluating the quality of intermediate reasoning steps rather than final outputs alone [177], reducing the incentive for agents to optimize for terminal answer appearance at the expense of reasoning integrity.

2.2 Robustness

Robustness is defined as the *"ability of a system to maintain its level of performance under a variety of circumstances"* (ISO/IEC TS 5723:2022; [3]), and represents a critical dimension of reliable AI. In machine learning, robustness is dissected along two axes: *non-adversarial robustness*, which concerns maintaining performance under naturally occurring variation such as distributional shift, noisy inputs, or domain generalization [14]; and *adversarial robustness*, which concerns resistance to deliberate perturbations designed to induce failure [14, 184]. Formally, robustness can be expressed as the difference in system correctness before and after perturbation, where a smaller gap indicates a more robust system [184]. Crucially, robustness and accuracy exist in tension meaning that models optimized for robustness often sacrifice performance on clean inputs, and vice versa [3, 6, 14].

In classical multi-agent systems, robustness extends beyond individual model performance to the collective ability of interacting agents to sustain coordinated behavior under stress [160]. This introduces a third axis absent in single-agent settings: *inter-agent robustness*, which concerns the system's ability to withstand failures originating from within, such as Byzantine agents, compromised communication protocols, and error propagation across agent boundaries.

In LLM-based MAS, all three axes manifest with heightened severity. Non-adversarial robustness is challenged by distributional shift in tool outputs, noisy API responses, and resource constraints that affect agent coordination [72]. Adversarial robustness faces a qualitatively new threat in the form of prompt injection attacks, where malicious inputs exploit the natural language interface of LLM agents to induce failures [37, 52, 72]. Most critically, inter-agent robustness is threatened by cascading failure dynamics. A single compromised agent can trigger a domino effect across the entire system [141], as adversarial content self-replicates through inter-agent communication rounds and propagates system-wide [178]. Empirical evidence further shows that MAS robustness is not an emergent property of model scale, but depends critically on agent roles, interaction design, and communication structure [158]. We organize the remainder of this section along these three axes.

2.2.1 Non-Adversarial Robustness. Non-adversarial robustness concerns the system's ability to sustain performance under naturally occurring variation, including distribution shift, noisy or failing tools, long-context degradation, partner variation, and resource constraints. A foundational challenge in LLM-based MAS is that failures are often silent. Agents may produce plausible outputs that pass superficial checks while carrying incorrect information forward through the pipeline [53]. This fail-open behavior becomes increasingly consequential in sequential workflows, where errors can compound across agents. Ao et al. [8] formalize this problem through a decision-theoretic model and show that a relay of agents cannot outperform a centralized Bayes-optimal decision maker unless individual stages introduce genuinely new information. In their MMLU experiments, a five-agent relay falls to 22.5% accuracy while a single agent achieves 90.7%. These findings show that additional coordination does not inherently provide robustness and may instead introduce new sources of failure.

Existing approaches address these failures through both adaptive coordination and workflow design. Reasoning Capacity can identify bottlenecks under changing task and resource conditions [111], while bounded-context decomposition can reduce failures caused by long contexts and noisy retrieval [189]. At the workflow level, dedicated validation agents and explicit checks before irreversible actions can prevent locally plausible errors from propagating downstream [53]. Selective activation, early stopping, and hierarchical scheduling further maintain output quality under limits on tokens, latency, and cost [76, 149]. These findings suggest that non-adversarial robustness is shaped as much by workflow and coordination design as by the capabilities of the underlying models.

2.2.2 Adversarial Robustness. Adversarial robustness concerns resistance to deliberate perturbations designed to induce failure. Although security is not a primary pillar of this survey, adversarial threats are relevant here when they degrade the reliable operation of the multi-agent system. LLM-MAS create additional attack surfaces because natural-language communication, tool integration, and multi-step execution allow adversarial content to propagate across agent boundaries [72]. Indirect prompt injection can spread from a poisoned retrieval event through subsequent interactions [52, 178], while compromising a high-authority or system-level role can affect downstream agents throughout a hierarchy [141]. Adversarial behavior can also originate within collaboration itself. A persuasive malicious participant can steer multi-agent debate toward incorrect conclusions [6], showing that mechanisms intended to improve collective accuracy can become channels for systematic manipulation.

Defenses operate at several levels, including adversarial stress testing, verification of peer reports, filtering of unreliable outputs, and aggregation mechanisms that limit the propagation of compromised contributions [59, 82, 132, 172]. Therefore, the reliability concern is not simply whether an individual agent resists an attack, but whether the collective can prevent an adversarial contribution from propagating through communication and aggregation. Detailed security threats and defense mechanisms are provided in Appendix E.

2.2.3 Inter-Agent Robustness. Inter-agent robustness concerns failures that originate within the agent population or emerge through interaction, including malicious or degraded agents, compromised communication, cascading errors, and sensitivity to network structure. These failures are particularly difficult to contain because an individual contribution may appear locally plausible while becoming harmful only after it is propagated or amplified by other agents. Early stage compromise can therefore have disproportionate downstream effects [141]. Insider agents present an even subtler threat. Sun et al. [137] show that an attacker can infer behavioral characteristics of its peers and exploit them to sustain disagreement while appearing cooperative, while Xie et al. [166] identify covert strategies such as reframing, fabricated evidence, and strategic delay. These findings show that inter-agent robustness requires monitoring the interaction process rather than evaluating agents only in isolation.

Topology and Communication Structure. Communication structure further determines how such failures propagate. Chen et al. [23] find substantial differences across multi-agent topologies under malicious agent injection, with shared-pool architectures more vulnerable than decentralized configurations and multi-round debate sometimes amplifying rather than suppressing risk. Adaptive routing and sparse communication provide complementary responses. AgentNet removes a centralized routing dependency, while GPTSwarm learns to prune low value communication links [171, 200]. These results establish topology as an active robustness mechanism rather than merely an implementation choice.

Aggregation Under Stress. Aggregation additionally determines whether local errors become collective failures. Majority and weighted voting can improve stability when agent outputs contain independent noise, while premature consensus can instead lock the group into an incorrect trajectory [81, 162]. Risk-aware aggregation addresses this problem through calibrated confidence, deferral, and escalation [74]. More broadly, open multi-agent systems can experience collective collapse when even a single participant alters the incentive or interaction structure [113]. Inter-agent robustness, therefore, depends jointly on detecting unreliable participants, controlling how information flows among them, and limiting the influence of unreliable contributions during collective decision making.

Additional implementation details concerning partner variation, resource constraints, attack and defense mechanisms, topology, aggregation, and architectural approaches are provided in Appendix E.

2.3 Resilience and Recoverability

While robustness concerns a system's ability to resist performance degradation when exposed to disturbance, resilience concerns what happens once degradation has already occurred. In general AI systems, resilience is defined as the capacity to detect a disruption or partial failure, adapt operation accordingly, and restore acceptable functionality over time, even if performance is temporarily reduced during the recovery process [3]. This distinction matters because a system can fail to be robust, in the sense that its performance drops under stress, while still being highly resilient, in the sense that it detects the drop and recovers quickly; conversely, a system can be robust to a narrow set of disturbances yet have no mechanism for recovery once it encounters something outside that set.

In multi-agent systems, resilience extends beyond the recovery capacity of any individual component to the collective ability of the system to reconfigure itself after a failure. This is most often achieved through decentralized designs and adaptive communication or routing strategies that allow the system to reroute around degraded or failed agents rather than depending on a single point of coordination [63, 171]. Or [108] give this capacity a quantitative basis by adapting classical dependability theory to cognitive orchestration, defining Mean Time-to-Recovery for Agentic Systems as the interval between detecting a reasoning or coordination fault and restoring coherent operation. Their accompanying Normalized Recovery Ratio relates this latency to fault frequency and is shown to lower-bound long-run cognitive uptime, offering a way to treat recovery speed itself as the measurable signature of resilience rather than relying on qualitative claims of adaptability. Resilience at this level is therefore a structural property of the agent network rather than a property of any single agent's reasoning.

In LLM-based MAS, resilience manifests across several layers of the system. At the interaction layer, Agashe et al. [2] provide evidence of zero-shot partner adaptation in cross-play settings without any retraining, showing that agents can adjust their coordination strategy when paired with a new and previously unseen partner, which reflects a recovery-oriented form of adaptability rather than mere resistance to partner variation. At the decision layer, Chen et al. [19] use multi-round debate with re-election when consensus fails, allowing the team to steer itself back toward a sound collective plan after an episode of disagreement rather than remaining stuck in a degraded state. At the control layer, Liang et al. [86] show that combining short-term and long-term memory with self-reflection allows agents to recover from earlier mistakes during a task, for example correcting a failed database operation by revising the approach based on the recorded outcome of the failure. Pezeshkpour et al. [111] formalize a Reasoning Capacity notion that locates weak links in a multi-agent pipeline under budget constraints, then applies self-reflection guided by human feedback to repair the components responsible for low capacity, restoring system performance through targeted and deliberate reconfiguration rather than through passive resistance to the underlying disturbance. The remainder of this subsection organizes prior work by the recovery mechanism each approach enables.

2.3.1 Rollback, Replay, and State Restoration. Versioned artifacts and execution logs make it possible to resume a pipeline from a known good state rather than restarting from scratch, which matters most when failures occur late in a long workflow. Mazaheri [96] operationalize this through REPOT, which deterministically replays a generated plan to identify its longest valid prefix, preserves that state as a checkpoint, and repairs only the remaining suffix. The checkpoint proves decisive: checkpoint-aware methods achieve over 70% recovery on Gemini compared with at most 3.1% for error feedback, confirming that knowing where execution was still correct is more valuable than knowing what went wrong. Related approaches use explicit system state and failure attribution to restart affected subtasks without discarding unaffected work [149, 188]. Fuller treatment of failure attribution appears in Section 3.1.

2.3.2 Corrective Loops and Memory. Recorded errors and self-reflection create internal feedback paths that allow a system to recognize when it has gone wrong and act before the problem compounds. Existing approaches use corrective re-prompts, reviewer feedback, memory, and dedicated monitoring agents to detect and repair faulty intermediate states [12, 63, 86, 172]. Shen et al. [129] extend correction to the multi-agent level through MASC, which detects erroneous outputs before they enter the shared interaction history and routes flagged steps to a correction agent in real time. Tamang and Bora [138] similarly use dedicated supervisors that monitor peer messages and intervene through quarantine, rerouting, or escalation. At production scale, Nanda et al. [103] show that asynchronous monitoring can provide course-correction guidance without blocking the main execution loop. These mechanisms follow the broader

MAPE-K control cycle [123], in which agents monitor state, analyze observations, plan a corrective response, and execute it over shared knowledge.

2.3.3 Consensus Repair and Team Adaptation. A wrong or unstable collective decision does not always require starting over; often the team can recover by changing who contributes and how agreement is reached. Liu et al. [90] pursue this through mid-run roster adaptation, removing agents whose contributions are dragging the team off course and adjusting the stopping rule so the pipeline pivots without a full restart. Wu and Ito [162] argue that premature convergence is itself a recovery risk, using implicit consensus to preserve enough reasoning diversity for the group to recover from an early wrong turn. Other approaches recover through renewed aggregation, negotiation, or re-election [19, 22, 81, 113]. These mechanisms restore collective performance by adapting participation and agreement rather than rebuilding the entire workflow.

2.3.4 Agent-level Routing and Structure Reconfiguration. Reconfiguring the collaboration or task graph during execution addresses failures at the level of how work is organized rather than how individual agents respond to errors. Yang et al. [171] allow agents to reroute tasks and reshape communication around successful paths, while Zhuge et al. [200] edit collaboration edges so that degraded channels can be bypassed. Guo et al. [54] extend dynamic graph repair through DynaGraph, which either inserts an auxiliary node to repair a localized gap or reconstructs an entire corrupted reasoning branch when broader structural repair is needed. Other approaches restructure task assignments, isolate risky agents, or selectively activate stronger participants under changing conditions [23, 76, 193]. These findings show that recoverability in LLM-MAS can depend on changing the structure of collective execution itself, particularly when failures cannot be repaired locally.

Additional implementation and evaluation details for these recovery mechanisms are provided in Appendix F.

3 Accountability

Accountability in AI systems concerns whether identifiable actors can be held answerable for the design, operation, and consequences of an AI system. It requires mechanisms for assigning responsibility, evaluating conduct against relevant obligations or standards, justifying consequential decisions, and enabling review, corrective action, or recourse when requirements are not satisfied [3, 13, 119, 157]. In this sense, accountability goes beyond identifying how an outcome was produced by connecting system behavior to responsible actors, applicable obligations, and mechanisms for review and remediation. In general AI, this requirement is already difficult to satisfy. Neural models produce outputs through computations that are not naturally decomposable into human-interpretable steps, and the absence of explicit reasoning traces makes it challenging to determine which inputs drove which outputs. Accountability frameworks developed for this setting, including auditing standards, documentation requirements, and impact assessments, largely treat the AI system as a single unit whose behavior must be explained and governed at the boundary between system and user [100, 119].

In classical multi-agent systems, accountability acquired a distinctly collective dimension that single-agent frameworks cannot address. When multiple agents interact to produce a joint outcome, responsibility cannot be straightforwardly assigned to any one participant, because each agent's actions are conditioned on the actions of others. Classical MAS research addressed this through normative frameworks that assigned role-based obligations and permissions to agents, formal causal responsibility models that quantified each agent's counterfactual contribution to an outcome [94], and coalition-level credit assignment methods such as the Shapley value [128] that decomposed collective outcomes into individual contributions [130, 160]. Many of these frameworks were formulated around agents with explicit roles,

action models, symbolic plans, or well-defined utility functions, which made it possible to reconstruct the decision process after the fact.

LLM-based MAS introduce a qualitatively different accountability problem. Agents reason through implicit computations rather than symbolic derivations, their internal reasoning states are not naturally available as explicit, faithful, and structured traces, and their decisions cannot generally be reconstructed through a formal proof or plan. At the same time, the system involves multiple interacting agents whose outputs influence one another through natural-language exchanges, tool calls, memory reads, and orchestration signals, making the attribution of any given outcome to a specific agent or input substantially harder than in either single-agent AI or classical MAS [17, 31]. Accountability gaps arise not only because individual agent reasoning is difficult to reconstruct, but because interactions among agents can produce emergent outcomes that no single agent intended or could have predicted [32].

Importantly, complete observability and auditability do not by themselves resolve this problem. A multi-agent trajectory may be fully recorded and each individual action may appear locally reasonable or compliant, while feedback loops, coordination patterns, aggregation mechanisms, or mutually reinforcing decisions nevertheless produce an undesirable collective outcome. In such cases, the difficulty is not recovering what occurred, but determining how responsibility should be assigned when the failure is distributed across interactions rather than attributable to a single agent or action. Execution records and provenance are therefore necessary evidentiary foundations for accountability, but they are not sufficient: they must be combined with methods that reason about distributed causal influence, interaction-level failures, collective obligations, and system-level responsibility.

Wang et al. [154] formalize a related process-level accountability gap, observing that evaluating LLM-based agents on final outputs alone reveals nothing about which evidence supported an intermediate claim, which agent message introduced an error, or which tool call produced a downstream consequence. Raza et al. [122] further situate this problem within broader governance requirements, noting that accountability must encompass not only technical traceability but also regulatory and organizational oversight.

Comprehensive accountability in LLM-based MAS must therefore address three interdependent dimensions. *Attribution* asks which agents, inputs, interactions, or reasoning steps contributed to a given outcome and to what extent. *Traceability* asks whether the execution process, including intermediate states, inter-agent messages, tool outputs, and relevant decisions, can be reconstructed after the fact. *Auditability* asks whether this evidence is sufficient for an independent reviewer to evaluate system behavior against applicable requirements and support corrective action or recourse. The remainder of this section examines how prior work in LLM-based MAS has approached each of these dimensions and where significant gaps remain.

3.1 Attribution

Attribution concerns precisely identifying which agent, input, or reasoning step was causally responsible for a given output or system state. It enables **responsibility assignment** by linking actions to specific roles or agent combinations, supporting targeted remediation when undesirable behaviors occur and providing the evidentiary basis for downstream accountability functions [3, 94]. In LLM-based MAS, this problem is substantially harder than in single-agent settings because outputs emerge from sequences of inter-agent exchanges, tool calls, and memory operations whose causal structure is not visible in final answers alone [17, 154]. Prior work approaches this challenge along two axes, namely what is being attributed (**credit** for contributions vs. **blame** for failures) and how attribution is measured through structural logging, marginal analysis, semantic tracing, behavioral monitoring, or causal graph methods. The following paragraphs organize prior work by measurement approach.

Structural Attribution: The most direct path to attribution is architectural. If every action, artifact, and state transition is linked to an identifiable agent at the moment it is produced, the system preserves evidence about who performed each part of the execution rather than requiring that information to be reconstructed after the fact. This establishes provenance and actor to action linkage, but does not by itself determine how much a recorded action causally contributed to the final outcome. AutoGen logs utterances, tool calls, and termination reasons against named roles, while ChatDev and CMAT preserve attribution through role-separated pipelines and explicit feedback responsibilities [86, 116, 161]. At larger scales, MegaAgent and ReSo maintain agent-linked artifacts, state transitions, and contribution records across more complex workflows [149, 193]. Wang et al. [154] formalize this idea through execution provenance, using a typed causal graph to connect outputs with the inputs, tool calls, memory operations, and agent messages involved in producing them. Structural attribution therefore provides the evidentiary foundation on which stronger causal, semantic, or failure-specific attribution can build.

Marginal and Counterfactual Attribution: Structural logging records what happened and who acted, but it does not quantify how much each agent's action shaped the collective outcome. Marginal and counterfactual methods address this by asking what would have happened had a particular agent or step been absent or different. The **Shapley value** [128] provides the foundational framework for distributing collective outcomes among individual contributors according to marginal contribution, while Mao and Gratch [94] extend related causal reasoning to MAS by modeling how relationships among agents determine individual responsibility for joint outcomes. Recent LLM-based MAS methods apply related counterfactual analysis to identify dominant or weak contributors, assign agent-specific rewards, and guide targeted correction [12, 33, 84, 111, 165]. These approaches provide a stronger basis for responsibility assignment than structural logging alone because they estimate how outcomes change when individual contributions are altered or removed.

Semantic and Content-Based Attribution: Where marginal methods ask what changes when an agent is removed, semantic methods ask what **informational value** each agent contributed within the trajectory that was actually executed. SLIC [69] formalizes this through Semantic Cooperative Games, tracing how agents generate, preserve, combine, or transform task-relevant semantic information and computing a Semantic Shapley Value that assigns credit according to informational contribution rather than structural position. This distinction is important because structural position and informational contribution need not coincide. An agent appearing late in a pipeline may contribute little new information while still having substantial power to alter the final output. Related Contribution and Credibility Scores estimate both an agent's contribution and the degree of influence it should receive during aggregation [42]. Semantic attribution therefore complements counterfactual analysis by examining what information an agent contributes, rather than only whether the outcome changes in its absence.

Behavioral and Influence Attribution: A third class of methods measures attribution through the observable dynamics of agent behavior and communication rather than through counterfactual intervention. MADC [187] finds that speaking order can exert stronger influence over collective decisions than answer correctness, with later-positioned agents often dominating final outcomes regardless of contribution quality. Pairwise influence analysis similarly shows how sycophancy can propagate through multi-agent discussions and reorganize influence across the team [71]. Other approaches estimate influence from contribution scores, task relevance, confidence, historical performance, or incentive structure and use these signals to regulate participation or aggregation [19, 61, 90]. These measures are useful for

identifying interaction patterns, but behavioral influence should not automatically be interpreted as causal responsibility when the underlying estimates depend on peer judgments, positional effects, or other behavioral proxies.

Failure Attribution and Benchmarks: Even with structural logging, marginal analysis, and behavioral monitoring in place, localizing the source of a failure remains a distinct and substantially harder problem. Failures in LLM-based MAS rarely trace to a single agent acting incorrectly. Cemri et al. [17] show that they often emerge from sequences of individually plausible decisions and from how agents communicate, coordinate, and propagate errors across the pipeline. Zhang et al. [188] formalize the problem by defining the **decisive error** as the agent-step pair (i^*, t^*) whose correction changes a failed run to success. Their benchmark reveals a substantial gap between agent-level and step-level localization accuracy, indicating that identifying the responsible participant is considerably easier than identifying the precise point at which the trajectory became decisive for failure.

A growing set of methods addresses this gap through richer execution traces, structured verification, constraint checking, and graph-based localization. TraceElephant shows that trace completeness substantially improves both agent and step attribution relative to output-only logs [24]. RAFFLES evaluates whether candidate steps are actual, early, and decisive faults [196], while AGENTRX derives and checks trajectory constraints to produce evidence-linked violation records [10]. GraphTracer and AgenTracer instead model attribution over structured dependencies among agents and execution steps [181, 183]. These approaches show that failure attribution increasingly depends on reconstructing the structure of interaction rather than inspecting isolated outputs.

A fundamental challenge is that failure attribution may be inherently **ambiguous**. MP-Bench reports substantial disagreement even among expert annotators over which steps should be considered responsible [65], suggesting that some failures admit several plausible attributions rather than a single ground truth culprit. Attribution becomes harder still when harmful behavior is intentional, as AgentXposed demonstrates through agents that appear cooperative while covertly sabotaging the team [166]. More broadly, attribution becomes less reliable as trajectories grow and long-range causal dependencies become harder to isolate, leaving the connection between diagnosis, verification, and repair incomplete [115]. These cases expose an important accountability gap. When responsibility cannot be assigned confidently to one agent or step, systems should preserve uncertainty over plausible causes rather than forcing a single attribution unsupported by the available evidence.

Additional implementation details, quantitative results, and method-specific mechanisms are provided in Appendix F.5.

3.2 Traceability

Attribution identifies which agent, component, or execution step contributed to an outcome, but such assignments require an evidentiary record of how that outcome emerged. Traceability provides this foundation by preserving the artifacts and dependencies needed to reconstruct system behavior. ***Traceability in AI systems*** refers to the extent to which a system's decisions, actions, and outputs can be reconstructed from the execution artifacts that produced them. These artifacts may include user instructions, prompts, model responses, retrieved evidence, intermediate outputs, tool calls and parameters, tool outputs, memory reads and writes, environmental observations, and final actions. Although structured logs establish a chronological record of what occurred, deeper traceability requires explicit relations showing how one artifact supported, generated, triggered, updated, contradicted, invalidated, or influenced another. Execution graphs and provenance representations therefore provide stronger traceability than flat interaction logs because they expose both the sequence of events and the dependencies underlying system behavior [5, 39, 133, 154].

In traditional multi-agent systems, traceability must additionally capture distributed execution across multiple autonomous entities. System-level outcomes may emerge from task delegation, coordination, information exchange, environmental interaction, and interdependent actions rather than from a single agent's decision. Traceability therefore requires recording agent identities and roles, local observations, exchanged messages, delegated tasks, actions, environmental state transitions, and the dependencies connecting local decisions to collective outcomes. Such records allow developers to reconstruct how information and responsibility-relevant evidence moved through the system, identify where coordination broke down, and determine how one agent's behavior influenced subsequent agents or the wider system state [34, 131].

LLM-based multi-agent systems introduce further complexity because agents communicate through natural language, generate stochastic outputs, use external tools, access shared or persistent memory, and repeatedly transform information across interaction rounds. ***Traceability in LLM-MAS*** should therefore support backward reconstruction from a final output, action, or failure to the relevant agents, intermediate claims, inter-agent messages, retrieved evidence, tool interactions, memory items, initial prompts, and contextual conditions. It should also capture communication specific information, including sender-receiver relationships, agent roles, delegation and routing decisions, message order, shared state updates, and verification status. When an orchestrator or intermediary agent summarizes, normalizes, filters, or rewrites a message, both the original and transformed versions should be preserved so that changes in intent, constraints, evidence, or safety-relevant meaning remain inspectable. Communication graphs, state histories, and tamper resistant audit logs can further reveal which agent introduced, propagated, verified, amplified, or failed to correct information that shaped the final outcome [72].

Existing approaches support traceability at progressively deeper levels. First, observability mechanisms preserve execution artifacts as structured records. Second, communication lineage connects these records across agent handoffs. Third, provenance models encode semantic and procedural dependencies among artifacts and, where supported, causal relations. These three layers support two operational uses: failure backtracking after execution and runtime intervention while execution is still in progress. We organize the following discussion around these three traceability layers and their two operational uses, while recognizing that individual systems may combine several of them [154].

Execution Artifacts and Structured Observability. The first requirement for traceability is a sufficiently complete and structured record of agent execution. Span based observability approaches capture planning, task execution, model invocation, tool use, evaluation, guardrail activity, and external interactions within a shared trace hierarchy. Common records include timestamps, parent and trace identifiers, inputs and outputs, model parameters, retrieved knowledge, task dependencies, errors, evaluation results, and intervention events [5, 39]. AgentOps develops a taxonomy of the artifacts and metadata that should be recorded across the agent lifecycle, whereas AgentTrace provides a runtime schema that organizes them across cognitive, operational, and contextual surfaces. Task specific frameworks similarly preserve intermediate outputs, supporting evidence, and verification results that connect final conclusions to earlier stages of execution [125, 189]. These artifacts should be understood as observable execution records and generated explanations rather than complete access to a model's internal reasoning process.

Inter-Agent Communication and Task Flow. Execution records become more informative when they show how information and tasks move between agents. Dialogue based systems preserve role conditioned exchanges, speaker order, and contextual responses, allowing a collaborative trajectory to be inspected turn by turn [11, 77]. More structured frameworks record agent identities and roles, speaker selection, messages, delegation decisions, tool calls, context updates, human interventions, and termination conditions [161]. Such records reveal which agent

initiated a communication or delegated a task, what information was available to the receiving agent, which routing or transformation occurred, and how the recipient responded. This detail is important because a failure may originate in the sender's output, an orchestrator's routing decision, an intermediary's transformation, or the recipient's interpretation. Preserving message identities, sender and receiver roles, delegation links, ordering, and transformed versions therefore supports reconstruction of information flow, task transfer, and responsibility relevant handoffs [72, 161].

Structural Provenance and Information Flow Tracing. Chronological records show which events occurred, but they do not necessarily establish which earlier artifacts supported later decisions. Structural provenance addresses this limitation by representing execution as a graph whose nodes correspond to agents, messages, prompts, model calls, tools, observations, memory items, intermediate claims, or outputs, and whose edges represent information flow and dependency. Workflow and provenance models therefore allow final outputs to be traced backward to the agents, prompts, data, tools, and intermediate results associated with their production [133, 200].

PROV-AGENT extends this representation through standardized provenance relations among agents, prompts, responses, model invocations, tools, decisions, workflow tasks, and domain data, supporting both backward tracing from an outcome to its inputs and forward tracing from an earlier artifact to later tasks and outputs [133]. Information flow methods extend this principle across semantic transformations, tools, memory, sessions, and agents. NeuroTaint, for example, tracks untrusted information from source events to privileged sinks even when the information is paraphrased, fragmented, or reused indirectly through persistent memory [16]. These approaches provide deeper traceability than chronological logging because they preserve how artifacts are connected, transformed, and propagated, and, where separately established, how one may causally affect another.

Post-hoc Failure Backtracking. Once a sufficiently complete trace is available, post hoc methods can use it to reconstruct where and how a failed trajectory departed from an acceptable execution path. Failure localization at the agent and step levels depends on access to the inputs, intermediate states, tool interactions, and environmental conditions surrounding individual actions [24, 188]. TraceElephant demonstrates the importance of this evidence directly. Compared with output only traces, fuller observability improves agent level attribution accuracy by 22% and step level accuracy by 76% [24]. Recent methods use these traces for temporal, semantic, procedural, and constraint based localization [10, 196, 198].

However, trace completeness does not guarantee a unique explanation of failure. The same execution may support responsibility claims concerning the agent that introduced an incorrect statement, an agent that amplified it, an orchestrator that routed it, or a verifier that failed to detect it. MP-Bench demonstrates substantial disagreement among expert annotators over failure localization [65]. Traceability should therefore preserve sufficient context to support evidence grounded alternative reconstructions rather than forcing a single causal interpretation. The attribution methods that operate over these records are discussed in greater detail in Section 3.1.

Runtime Monitoring and Provenance-Aware Intervention. Trace information can additionally support monitoring and intervention while execution is still in progress. Runtime supervision methods associate deviations with specific agents, messages, and actions and record the interventions taken in response [138, 166]. Provenance aware defenses extend this principle to the origins of tool arguments and the authorization boundaries governing agent actions [126, 147]. Agent-Sentry records the sources of values used in tool calls and how retrieved information contributes to sensitive actions, while AUTHGRAPH compares the realized execution trajectory with a separate representation of the

tools and information flows authorized by the original request. In both cases, trace information becomes an operational signal for deciding whether an action should proceed, be blocked, or receive additional review.

These approaches illustrate a progression from passive recording to active use of trace information. After execution, traces support reconstruction and failure diagnosis. During execution, the same artifacts and dependencies can support blocking, correction, rerouting, or escalation before an error propagates. Therefore, traceability provides the evidentiary infrastructure needed for failure diagnosis, responsibility analysis, and subsequent review of system behavior. Additional implementation details for observability, communication lineage, provenance, failure backtracking, and runtime tracing are provided in Appendix F.6.

3.3 Auditability

Auditability in AI systems refers to the extent to which a system produces trustworthy and sufficiently complete evidence that allows its behavior, decisions, and compliance to be independently evaluated against stated requirements, policies, standards, and claims. Auditability is a property of the system, whereas auditing is the process of examining the available evidence. Accountability is the broader objective that includes determining whether requirements were satisfied, assigning responsibility when they were not, and supporting review, remediation, or recourse. Auditability therefore provides the evidentiary basis for auditing and accountability [100, 105, 119].

Producing logs alone does not make a system auditable. Audit evidence should support five complementary capabilities: *action reconstruction*, which enables reconstruction of policy-relevant actions and their inputs, parameters, outputs, actors, and timestamps; *lifecycle coverage*, which includes planning, delegation, approvals, retries, fallbacks, interventions, and termination; *policy checkability*, which allows compliance rules to be evaluated through repeatable or automated procedures; *attribution support*, which preserves evidence linking the acting component to its upstream delegation, authorization, or influence chain; and *evidence integrity*, which protects records against alteration, deletion, or fabrication [105]. These capabilities are jointly important because even complete execution records may be unsuitable for auditing when policies cannot be checked, responsibility cannot be established, or the evidence cannot be trusted.

In traditional multi-agent systems, auditability must cover both individual agents and the collective process through which system behavior emerges. Relevant evidence includes agent identities, roles, permissions, commitments, local observations, exchanged messages, delegated tasks, actions, environmental changes, and coordination rules. Audit records should allow reviewers to determine whether agents acted within their authority, whether responsibilities and tasks were transferred correctly, whether obligations were fulfilled, and whether the collective process complied with system-level requirements. Because evidence is distributed across agents and components, consistent identifiers, synchronized records, preserved delegation relations, and protections against selective deletion or modification are also required [34, 105, 131]. Multi-agent audit evidence should cover the full execution lifecycle rather than only the final action. A failure may originate from an earlier delegation, rejected approval, ignored warning, unsuccessful retry, inappropriate fallback, or downstream use of incomplete information. Records should therefore preserve completed and rejected actions, delegation and authorization changes, recovery attempts, human interventions, and other events that affect how decision authority and responsibility move through the system [105].

Auditability in LLM-based multi-agent systems must additionally account for natural-language communication, stochastic generation, dynamic agent selection, changing communication structures, retrieval, tool use, shared or persistent memory, and interaction with external environments. An auditable LLM-MAS should preserve versioned evidence of system and role prompts, model and decoding configurations, agent identities, inter-agent messages, delegation and routing decisions, retrieved sources, tool calls and parameters, memory operations, policy checks,

human interventions, and final outputs or actions. When an orchestrator or intermediary agent summarizes, filters, or rewrites another agent's message, both the original and transformed versions should remain available so that changes in evidence, permissions, constraints, or meaning can be examined [5, 39, 72, 105, 133].

The evidence and access required for an audit depend on its scope. Governance audits examine organizational policies, responsibilities, risk ownership, documentation, and oversight. Model audits examine development procedures, evaluation results, behavioral properties, and limitations. Application audits examine how models are configured and used in a deployed system [100]. In LLM-MAS, application audits should include the interaction mechanisms through which agents exchange information, delegate authority, use tools, and influence collective outcomes. Governance and model audits may require access to internal documentation, procedures, and personnel, whereas some application audits may be conducted through external testing and observed system behavior.

The existing literature addresses different stages of the audit process. Conceptual frameworks specify the evidence and access conditions needed for an audit, execution frameworks produce reviewable records, audit protocols evaluate those records against stated requirements, runtime mechanisms assess behavior during operation, and post-hoc methods support incident investigation. These functions are related but should not be treated as interchangeable, since recording an event does not by itself establish whether the event complied with a policy or whether the resulting evidence can support an independent audit.

Evidence Capture and Visibility: Visibility mechanisms establish which agents and responsible actors can be associated with consequential behavior. Agent identifiers, activity logs, real-time monitoring, and agent cards can document a deployed agent's goals, permissions, tools, interactions, and responsible stakeholders [18]. Agent frameworks provide more detailed operational evidence by recording conversations, speaker selection, tool interactions, routing and filtering decisions, task dependencies, monitoring results, and termination conditions [76, 149, 161, 171]. These mechanisms support action recoverability, lifecycle coverage, and attribution support by showing what agents received, produced, selected, rejected, or transferred. They primarily produce evidence, however, and do not independently determine whether the recorded behavior complied with an external requirement.

Audit Protocols and Independent Evaluation: Audit protocols make recorded behavior assessable by defining evaluation criteria, evidence requirements, procedures, and reporting formats. FACT-AUDIT specifies evaluator roles, graded evidence, an adaptive review process, and quantitative measures for assessing factuality and rationale quality [87]. Machine-checkable tests similarly translate structural requirements, role boundaries, output formats, and confidentiality constraints into conditions that can be applied consistently to agents and interaction turns [49]. HarnessAudit extends this principle to the complete agent trajectory by treating the execution harness, including its allocation of tools, resources, permissions, and communication channels, as the unit of evaluation [89]. Its results show that successful task completion does not necessarily imply compliant execution, since agents may reach the requested outcome while violating permission, resource, or information flow boundaries. Human-in-the-loop frameworks complement automated checks when audit criteria require contextual or qualitative judgment [7].

Continuous Auditing and Intervention: Audit checks may also operate while a trajectory is unfolding. Constraint State Governance represents safety requirements as explicit, versioned constraints and checks authority, action scope, information flow permissions, and supporting evidence before admitting critical actions. It records the active constraint, proposed action, delegated capability, decision, and justification in a hash-linked audit trail [83]. This approach connects

policy verifiability with evidence integrity and prevents constraints from being weakened as they pass through communication, delegation, memory, or tool use.

Other systems use runtime findings to **initiate intervention**. AgentForesight identifies suspected decisive steps and responsible agents before errors propagate further, while Enforcement Agents and MedSentry link screening, verification, isolation, and escalation decisions to the behavior that triggered them [23, 138, 179]. EnviSmart applies related principles in a production workflow by separating worker, validation, publication, and orchestration roles and requiring deterministic checks before irreversible actions [53]. Together, these approaches show that audit evidence can support timely containment and correction rather than serving only post-hoc review. Post-hoc failure localization can likewise contribute evidence to an audit, although identifying the agents or steps responsible for failure is primarily an attribution problem and is discussed in Section 3.1.

Across these categories, ***evidence integrity and controlled access remain common requirements***. Logs, evaluation results, intervention decisions, and incident findings cannot support a defensible audit if they can be altered, selectively removed, or fabricated. Versioning, append-only storage, cryptographic signatures, hash-linked records, access controls, redaction, and retention policies can protect audit evidence while limiting the exposure of private prompts, retrieved documents, tool parameters, credentials, and memory contents [1, 3, 83, 105]. Additional implementation details for evidence capture, audit protocols, and runtime auditing are provided in Appendix F.7.

4 Transparency and Understandability

Transparency and understandability in AI systems concern whether relevant information about a system is both available to appropriate stakeholders and presented in a form that enables them to make sense of its behavior. *Transparency* concerns the availability and visibility of information about the system, including its intended purpose, inputs and data sources, configuration, decision processes, outputs, capabilities, and limitations. *Understandability* concerns whether stakeholders can use this information to form a meaningful understanding of how the system operates and why particular behaviors or outcomes arise. The two properties are therefore closely related but distinct: making information available does not necessarily make a system understandable, while meaningful understanding generally requires access to relevant information. Information requirements may also vary across stakeholders and purposes [3].

In traditional multi-agent systems, transparency and understandability extend beyond the behavior of individual agents to the distributed processes through which collective behavior emerges. Relevant information may include agent identities and roles, local observations, assigned tasks, exchanged messages, coordination rules, intermediate decisions, and sequences of actions [130, 160]. However, making these elements visible does not by itself explain the resulting collective behavior. Because information, authority, and decision-making are distributed across multiple agents, stakeholders must also be able to understand how local actions and interactions combine to produce a system-level outcome. Transparency and understandability in MAS therefore concern both visibility into individual agents and the ability to make sense of the communication, coordination, and dependencies connecting their behaviors.

In LLM-based multi-agent systems, these requirements become more challenging because system behavior develops through stochastic language generation, multi-step planning, dynamic delegation, retrieval, tool use, persistent or shared memory, orchestration, and environmental feedback. An explanation of one agent or one decision may therefore be insufficient to understand the resulting system behavior. An earlier message, retrieval result, memory update, or planning decision may alter the context available to later agents and become consequential only several steps later. Recent work accordingly argues that interpretability methods developed for static input-output models provide

only component-level explanations and do not adequately capture the temporal, interactive, and context-dependent behavior of agentic systems [17, 197].

Transparency and understandability in LLM-MAS consequently need to be supported at three connected levels. At the *agent level*, they concern the inputs, outputs, relevant reasoning artifacts, capabilities, limitations, and actions of individual agents, as well as whether these elements can be meaningfully related to the agent's behavior. At the *interaction level*, they concern inter-agent communication, delegation, role assignment, coordination, routing, and the information available to agents when decisions are made. At the *system level*, they concern how memory and state evolve over time, how tools and environmental feedback affect subsequent decisions, how information or errors propagate across agents, and how local behaviors combine into collective outcomes [58, 122, 197]. Agent identifiers, activity records, and monitoring mechanisms can provide visibility into participation, interactions, tool use, and state changes [18], while explanation and interpretation mechanisms help stakeholders make sense of how these observations relate to system behavior.

Importantly, neither transparency nor understandability should be equated with simply exposing generated reasoning. Natural-language rationales may appear understandable without faithfully representing the factors that produced an agent's decision, while detailed execution records may make system activity visible without making the resulting behavior understandable. Moreover, component-level explanations may fail to capture dependencies that develop across a complete multi-agent trajectory [197]. Meaningful transparency and understandability therefore require both sufficiently reliable information about system behavior and mechanisms that help stakeholders interpret this information while recognizing the uncertainty and limitations of what it can establish.

We characterize transparency and understandability through three complementary dimensions: *explainability and interpretability*, *uncertainty and limitations*, and *reproducibility*. *Explainability* concerns how the basis of outputs, actions, or collective behaviors is communicated to relevant stakeholders, while *interpretability* concerns whether stakeholders can meaningfully understand the factors shaping system behavior. *Uncertainty and limitations* concern the communication of confidence, ambiguity, assumptions, and capability boundaries, whereas *reproducibility* concerns whether sufficient information is available to examine behavior across repeated executions. Together, these dimensions capture the information needed to make LLM-MAS behavior both observable and understandable.

4.1 Explainability and Interpretability

Explainability and interpretability are closely related requirements for making AI behavior understandable, although their precise definitions and boundaries vary across the literature [88, 93, 102, 190]. We use *explainability* to refer primarily to mechanisms that communicate how an output, action, or decision was produced, and *interpretability* to refer to the extent to which relevant stakeholders can make sense of system behavior and the factors that shape it. Because these objectives substantially overlap in practice, particularly in LLM-based multi-agent systems, we examine them jointly rather than treating them as strictly separable dimensions.

Traditional explainability and interpretability methods primarily focus on individual models and predictions. They include intrinsically interpretable models and post-hoc techniques such as feature attribution, counterfactual explanations, probing, concept-based explanations, and natural-language rationales. Such explanations may target a particular prediction or provide a broader account of model behavior [93, 102, 190]. Mechanistic interpretability provides a complementary white-box perspective by identifying internal features, neurons, attention heads, or circuits and examining how they contribute to model behavior [51]. For LLMs, generated rationales and Chain-of-Thought can make intermediate reasoning more legible to users, but their explanatory value depends on *faithfulness*: whether the

explanation accurately reflects the factors or processes that actually produced the model's output. A rationale may be coherent and persuasive while omitting, misrepresenting, or even contradicting the information that influenced the underlying decision. Generated reasoning should therefore be treated as an explanatory artifact rather than direct evidence of a model's internal computation [102, 197].

In LLM-based multi-agent systems, the object of explanation expands from an individual prediction to a distributed and temporally evolving execution. Collective outcomes may depend on which agents participated, what information they received, how messages were transformed or interpreted, how tasks were delegated, how memory and tools affected later decisions, and how disagreements or intermediate errors propagated through the system. Consequently, explaining individual agents in isolation may be insufficient even when each local decision appears understandable. Recent work therefore argues for *system-level interpretability*, in which explanations account for the interactions and temporal dependencies through which collective behavior emerges [107, 197]. Existing work addresses this problem through several complementary approaches.

Mechanistic Interpretability in LLM-MAS:. Mechanistic interpretability may complement system-level approaches by examining the internal computations of individual agents at behaviorally important points in a multi-agent trajectory. For example, once interaction- or trace-level analysis identifies an agent message or decision that strongly affected a collective outcome, mechanistic methods could probe the internal features, attention heads, or circuits associated with that local behavior. This creates a possible connection between system-level analysis, which identifies *where* consequential behavior emerged, and mechanistic interpretability, which investigates *how* the underlying model produced that behavior. However, mechanistic analysis of individual agents does not by itself explain collective dynamics such as delegation, persuasion, information propagation, or emergent coordination, and applying it across long multi-agent trajectories remains computationally and methodologically challenging [51, 197].

A first class of approaches grounds explanations in artifacts produced during execution rather than relying solely on agents to describe their own behavior. These artifacts may include intermediate outputs, agent states, messages, tool interactions, contextual information, confidence estimates, and decision records. Such evidence can then be transformed into a human understandable account of how a collective outcome developed. Work on agentic interpretability describes this as *trace-based interpretability*, where inter-agent interactions provide evidence from which the development of a collective decision can be reconstructed [107]. Related work emphasizes that system-level interpretation requires connecting information across reasoning, planning, memory, actions, and interactions rather than examining individual components independently [197]. This illustrates an important distinction between traceability and explainability: execution traces preserve evidence of what occurred, whereas explanation mechanisms organize and communicate that evidence so that stakeholders can understand its significance.

Interaction- and System-Level Interpretation. Other approaches focus directly on collective dynamics that cannot be understood from individual-agent outputs alone. SwarmProbe analyzes the evolution of agent stances across interaction rounds to expose consensus formation, polarization, leader emergence, and turning points in collective decision making [195]. Its influence measure captures temporal association between messages and subsequent stance changes rather than causal responsibility, but the resulting representation allows users to inspect how group behavior develops over time. Complementary data-centric approaches identify higher-level behavioral concepts and interaction patterns across multi-agent trajectories, including differences between successful and failed runs and emerging behaviors that may not yet be visible in aggregate performance measures [170]. Together, these approaches move interpretability

from explaining isolated decisions toward identifying the behavioral structures and interaction patterns that characterize the system as a whole.

Fine-Grained Explanations of Agent Behavior. System-level interpretation can be complemented by explanations that identify which parts of an agent's observable behavior contributed to a particular system judgment. XG-Guard, for example, detects compromised agents using both sentence- and token-level representations of multi-agent communication and provides token-level explanations indicating which portions of an agent's message contributed to its anomaly score [109]. Since these explanation scores are derived from the same anomaly-scoring mechanism, they are directly tied to the detector's decision rather than generated independently as a natural-language rationale. Such explanations nevertheless remain local: they explain why a particular agent was flagged in the current dialogue context rather than how the complete multi-agent trajectory produced its final outcome. This illustrates the complementary roles of fine-grained agent-level and broader interaction-level explanations.

Interactive and Human-Centered Explanations: The appropriate form and level of explanation depend on the intended audience, its expertise, and the purpose for which the explanation is provided. *Developers and system operators* may require detailed information about agent interactions, intermediate states, tool use, and failure propagation to debug or improve a system. *End users* may instead require concise and accessible explanations of the main factors that shaped an outcome, together with relevant uncertainty and limitations. *Auditors or regulators* may require structured evidence connecting consequential decisions and actions to the agents, information, controls, and requirements involved. Explanation quality should therefore be assessed relative to the information needs and technical expertise of the intended stakeholder rather than assuming a single explanation is appropriate for all audiences.

Interactive approaches provide one way to accommodate these differences by allowing stakeholders to request progressively more detailed information. LLMCheckup demonstrates this principle for individual LLMs, while XAgen extends it to multi-agent workflows by transforming execution records into an interactive representation of agents, tasks, tools, intermediate outputs, and execution order [148, 152]. These systems illustrate a shift from static, uniform explanation artifacts toward interactive explanation processes in which stakeholders can explore system behavior at a level appropriate to their role and task.

Faithfulness and Evaluation: The faithfulness problem becomes more difficult in LLM-MAS because collective outcomes may depend on information that is repeatedly communicated, summarized, transformed, or acted upon by different agents. An explanation may therefore appear coherent while omitting interactions that materially shaped the trajectory. Likewise, natural-language rationales may not accurately reflect the factors that produced an agent's decision, feature-attribution methods may identify influential signals without fully characterizing the underlying decision process, and system-level summaries may abstract away dependencies among agents, tools, memory, and environmental feedback [93, 102, 197]. Explainability and interpretability should therefore be evaluated not only by whether an explanation is understandable, but also by whether it is adequately grounded in the evidence and behavior that produced the outcome.

Following established interpretability literature, explanation evaluation can be considered along *functionally grounded*, *human-grounded*, and *application-grounded* criteria [93]. ***Functionally grounded*** evaluation concerns whether an explanation faithfully reflects the behavior of the system being explained. In LLM-MAS, this can be examined by perturbing or intervening on agents, messages, features, or interaction steps identified as important and measuring whether the expected change in system behavior occurs. ***Human-grounded*** evaluation concerns whether explanations

are useful and understandable to intended users, using measures such as comprehension or analysis accuracy, error-identification accuracy, completion time, confidence, or perceived interpretability. ***Application-grounded*** evaluation assesses explanations in the task or environment for which they are intended, such as debugging, failure localization, supervision, intervention, or system refinement.

Recent LLM-MAS work illustrates these complementary forms of evaluation. SwarmProbe evaluates system-level explanations through users' analysis accuracy, completion time, and confidence when identifying collective behavior [195]. XAgen evaluates whether workflow visualizations and error explanations improve users' understanding and support diagnosis of multi-agent executions [152]. Yan et al. [170] combine human ratings of interpretability with behavioral validation, testing whether discovered hypotheses help users distinguish training stages and whether injecting those hypotheses back into an agent changes downstream performance. These examples demonstrate that explanation quality cannot be captured by a single metric: an explanation may be understandable but poorly grounded in the underlying process, or faithful to system behavior but too complex to support its intended user or task. Additional implementation details and empirical results for these explanation and interpretation approaches are provided in Appendix F.8.

4.2 Uncertainty & Limitations

Uncertainty and limitations concern whether an AI system can represent and communicate uncertainty about its information, reasoning, actions, and outcomes, as well as recognize conditions under which available information or system capabilities are insufficient for reliable autonomous action. *Limitations* refer to these capability, knowledge, or operating boundaries that constrain when the system can be expected to perform reliably. Uncertainty may arise from incomplete or ambiguous inputs, model reasoning, environmental variability, tool interactions, or other components involved in system execution. Making these uncertainties and limitations visible is important because a system may produce a plausible output even when the evidence supporting it is weak or incomplete.

In LLM-based multi-agent systems, uncertainty is not solely a property of an individual agent or its final output. It can evolve as agents exchange information, aggregate evidence, use tools, update state, and make sequential decisions. An uncertain intermediate result may be communicated to another agent, transformed into a new representation, or reused in a later decision, causing uncertainty to be reduced, amplified, obscured, or propagated through the system. Recent work therefore studies uncertainty at multiple levels, including individual generations, agents, interaction rounds, trajectories, and the system as a whole [164, 186, 191].

Transparency and understandability consequently require more than exposing a confidence score at the final output. Relevant uncertainties should be identified and represented at the points where they arise, preserved or communicated when they affect downstream decisions, and related to the system's operating boundaries. When uncertainty becomes consequential, the system should also support appropriate responses, such as seeking additional information, requesting clarification, performing further verification, adapting its behavior, reducing autonomy, or escalating a decision for human review [44, 164, 186]. We organize existing work around three complementary capabilities: *uncertainty estimation and representation*, *uncertainty propagation and evolution*, and *uncertainty-aware response and limitation awareness*.

Uncertainty Estimation and Representation: A first challenge is determining what collective uncertainty means when an outcome depends on several agents. Confidence reported by individual agents provides only a partial view, since uncertainty may also arise from disagreement between agents, dependence created through communication, or instability across multiple executions of the same collaborative process. Recent work therefore constructs system-level

uncertainty estimates from several complementary signals. Collaborative Entropy separates uncertainty arising within individual models from disagreement across models, allowing shared ambiguity to be distinguished from cases in which individually confident agents support conflicting answers [135]. DISCOUQ similarly shows that vote counts alone provide an incomplete measure of collective confidence and instead considers the structure and evidence underlying disagreement [67]. The reliability of agreement becomes still harder to assess after communication because agent errors are no longer independent. CAGE-CAL explicitly models this dependence and identifies *communication-induced over-confidence*, where interaction drives agents toward the same incorrect answer and makes agreement appear more reliable than it is [62]. These findings show that collective uncertainty depends not only on confidence or consensus, but also on the source of disagreement, dependence among agents, and stability of the collaborative process.

Other approaches characterize uncertainty from variation across the trajectory or across model configurations. MATU measures departures from recurring patterns across agents, reasoning steps, communication structures, and repeated executions [25]. Related multi-model work separates within-model uncertainty from disagreement across heterogeneous LLMs and shows that diversity is useful only when disagreement reflects complementary information rather than noisy predictions [73]. Alternative query formulations can also expose uncertainty that repeated answers to the same prompt may fail to reveal [48]. These approaches broaden uncertainty estimation beyond individual confidence toward the structure, source, and stability of collective reasoning.

Uncertainty Propagation and Evolution: Estimating uncertainty at one point in an execution does not ensure that the same uncertainty remains visible or retains the same meaning later in the trajectory. When information crosses component or agent boundaries, its uncertainty may be reformulated, stored, consumed by a control mechanism, or communicated to another agent [164]. Handoffs, control decisions, and persistent state are therefore particularly important because uncertainty may change in meaning or consequence at these points. Related work proposes maintaining explicit uncertainty records that track source, supporting evidence, confidence, risk, dependencies, and lifecycle state as execution proceeds [186].

The loss or distortion of uncertainty across agent boundaries has also been demonstrated empirically. Emde et al. [46] describe *vanishing uncertainty*, where information passed from one agent to another is relayed with substantially less visible uncertainty than was present at its source. Repeated interaction can produce a different failure. Entropy and disagreement may persist across rounds rather than being resolved, and additional collaboration does not necessarily improve accuracy [191]. In collective hallucination settings, repeated peer adoption can make an unsupported claim increasingly dominant while agents become more confident in the resulting incorrect consensus, with network topology affecting whether the hallucination diminishes or spreads [66]. These findings show that communication, coordination, topology, and repeated interaction may suppress, amplify, transform, or conceal uncertainty over time. Increasing agreement therefore does not necessarily indicate that uncertainty has been successfully resolved.

Uncertainty-Aware Response and Limitation Awareness: Uncertainty becomes operationally useful when it influences what the system does next. Depending on its source and consequences, a system may seek additional evidence, ask for clarification, invoke verification, involve additional agents, modify coordination, abstain from acting, reduce its level of autonomy, or escalate a decision for human review. The framework of Zhang et al. [186], for example, connects changing uncertainty states to adaptations including specialist or critic agents, additional verification, changes to interaction structure, restricted autonomy, and human intervention, although these mechanisms are presented primarily as a system engineering framework rather than as a quantitatively validated adaptation policy.

More targeted work shows how uncertainty can trigger particular actions. Edwards and Schuster [44] separate task execution from ambiguity detection so that a specialized agent can require clarification when important information is missing. iMAD uses uncertainty-related signals to determine whether an initial answer should be retained or passed to multi-agent debate, avoiding the assumption that additional interaction is always beneficial [47]. Other methods use uncertainty during collaboration to control how strongly agents respond to one another, for example by weighting peer contributions according to confidence or conditioning belief revision on the confidence associated with received information [41, 199]. Uncertainty can therefore serve not only as information reported after a decision, but also as a coordination signal that determines when interaction should occur and how much influence individual contributions should receive.

Uncertainty may also indicate when the system should refrain from committing to an answer. Variation across agents or query contexts can expose cases in which the underlying knowledge is insufficiently reliable for autonomous action [48]. Across the literature reviewed here, *limitation awareness* is generally implemented through narrower signals such as missing information, disagreement, unstable reasoning, insufficient confidence, or the need for additional verification. These mechanisms allow a system to recognize particular conditions under which autonomous execution should be modified or interrupted. The ability to reliably identify and communicate broader competence boundaries across tasks and operating conditions remains comparatively underexplored in LLM-based multi-agent systems.

Additional implementation details and quantitative results for uncertainty estimation, propagation, and uncertainty-aware response are provided in Appendix F.9.

4.3 Reproducibility

Reproducibility in AI and Machine Learning. *Reproducibility* in AI and machine learning concerns whether reported results can be independently obtained when the relevant experimental artifacts and conditions are made available. The term is sometimes used inconsistently across the literature. We follow the distinction in which *repeatability* refers to obtaining the result again by the original researchers under the original setup, *reproducibility* refers to an independent team obtaining the result using the original code and data, and *replicability* requires the result to hold under an independently constructed implementation or dataset [118]. Reproducibility therefore depends on sufficient visibility into the experimental process, including the implementation, data preparation, model configuration, hyperparameters, software dependencies, and evaluation procedure. In complex AI pipelines, a high-level description of the method may not be sufficient since seemingly minor implementation or environment choices can materially change the result [55].

Reproducibility in LLM-Based Agents. For LLM-based agents, reproducibility becomes more difficult because execution is shaped not only by the underlying model but also by prompts, decoding settings, model versions, tools, external services, software dependencies, and the state of the surrounding environment. These factors may affect both whether an agent can be executed again and whether a repeated run produces comparable behavior. Work on coding agents, for example, shows that generated artifacts may fail to execute in a clean environment when required dependencies or execution assumptions are not adequately specified [144]. Other work addresses this problem by making agent configurations and evidence more explicit. AgentHub records version-specific information about capabilities, dependencies, environments, permissions, and evaluation procedures so that claims about an agent can be rerun and checked [110]. AgentReport takes a more controlled approach by fixing agent roles, execution order, data partitions, random seeds, and generation settings to reduce variation across repeated executions [29]. These examples suggest that

reproducibility in agentic systems concerns both preserving the conditions needed to execute the system and making enough of its operational configuration available for others to examine and repeat the process.

Reproducibility in LLM-Based Multi-Agent Systems. In LLM-based multi-agent systems, these requirements extend to the interaction structure through which collective behavior is produced. A reproducible description may need to specify not only the individual models and prompts but also agent roles, communication rules, coordination mechanisms, interaction order, shared state, tool access, and the conditions under which information moves between agents. Variability introduced at any of these points can alter the collective trajectory even when the nominal task and system architecture remain unchanged. This creates an important distinction between reproducing a final performance score and reproducing the conditions and interactions that produced it.

Reproducible Evaluation and Execution. Kaliyev and Maryanskyy [70] show that apparent coordination improvements can fall within ordinary run-to-run variation and may fail to persist across seeds, motivating repeated matched evaluations before small gains are attributed to the multi-agent design. Reproducibility also depends on preserving enough execution information to determine which parts of a workflow must be rerun when the system changes. Execution-lineage approaches address this by recording explicit dependencies among intermediate artifacts and selectively recomputing only the affected parts of an agentic workflow [124]. More broadly, evaluations of agentic systems increasingly call for reporting exact model versions, prompts, generation settings, and execution trajectories so that experimental results can be independently examined and compared [80]. Hence, we view reproducibility in LLM-MAS as the ability to preserve and disclose sufficient system, execution, and evaluation information for independent researchers to rerun the system under comparable conditions and determine whether its reported behaviors and findings persist.

5 Fairness & Ethics

Fairness and Ethics in AI and Machine Learning. *Fairness* in AI and machine learning concerns whether a system creates unjustified disparities in the treatment, opportunities, resources, services, or outcomes experienced by individuals or groups. The literature offers multiple notions of fairness, including individual and group-based criteria, and no single formulation is appropriate for every application [97]. The choice of what constitutes a relevant disparity depends on the affected population, the type of harm under consideration, and the context in which the system is used. Fairlearn therefore treats fairness as a sociotechnical concern and distinguishes, among other cases, harms arising from the allocation of opportunities or resources from differences in the quality of service received by different groups [155]. In this survey, we use *ethics* more broadly to refer to whether the behavior of an AI system remains consistent with normative requirements governing acceptable conduct, including requirements related to harm, discrimination, privacy, autonomy, integrity, and appropriate use of authority. Fairness is thus an important part of ethical AI, while ethical concerns extend beyond the distributional or group-based disparities usually studied under fairness [98].

Fairness and Ethics in Agentic and Multi-Agent Systems. The scope of these concerns expands when AI systems move from producing isolated predictions to pursuing goals through sequences of decisions and actions. An agent may retrieve information, use tools, allocate resources, or act on behalf of a user before a final response is produced. Evaluating only the terminal outcome may therefore overlook disparities or norm violations that occur during execution. Multi-agent systems add a further difficulty because outcomes depend on interactions among multiple decision makers. Traditional multi-agent research has consequently studied fairness in settings where agents compete for resources

or pursue different utilities, including objectives that protect the least advantaged participant while retaining overall system efficiency [180]. Other work shows that fairness objectives can be incorporated into local agent policies and coordinated through decentralized interaction rather than imposed only on the final collective outcome [68]. These settings make clear that fairness may concern not only what the system ultimately produces, but also how benefits and burdens are distributed among participants and how individual and collective interests are balanced.

The move to language-based agents also makes the normative basis of fairness more important. A recent review of fairness in multi-agent AI finds that many studies do not clearly identify the beneficiary of fairness or the normative principle that the system is expected to satisfy. In some cases, fairness is effectively delegated to foundation models through general instructions to behave fairly, without specifying what fair behavior means in the application being studied [4]. The same review identifies interaction-level concerns including herding, collusion, fairness drift, and the amplification of unfair norms. Fairness in an agentic system therefore requires more than assigning a fairness-oriented role or prompt to an individual agent. The relevant stakeholders, possible harms, and expected standards of conduct must be made sufficiently explicit to evaluate whether the resulting behavior is fair or ethically appropriate.

Fairness and Ethics in LLM-Based Multi-Agent Systems. LLM-based multi-agent systems make this distinction particularly important because properties observed for individual agents do not necessarily carry over to the collective system. Communication can change the biases and preferences expressed during execution. Nguyen et al. [104] show that communication can change the bias profile of the group itself, with previously unexpressed stereotypes appearing during interaction and becoming increasingly prevalent across agents over subsequent rounds. Similarly, Okawa [106] show that relatively weak individual biases can develop into strongly biased collective consensus through conformity and repeated interaction. Their analysis also indicates that agent heterogeneity and interaction conditions affect whether such biased consensus becomes established. More broadly, interaction topology has been identified as a source of variation in safety and fairness because ordering, connectivity, and aggregation determine which agents and viewpoints receive greater influence during collective decision-making [9]. These findings show that evaluating the underlying models independently is insufficient for assessing the fairness of the multi-agent system that they form.

Fairness in LLM-MAS also concerns the process through which agents treat and respond to one another. For example, interactional fairness evaluates whether communication is respectful and whether decisions are accompanied by adequate and understandable justification. Empirical work shows that these properties can affect how agents respond to the same allocation even when the allocation itself remains unchanged [11]. Fairness can therefore be affected both by the distribution of an outcome and by the manner in which the collective process is conducted. Diversity can likewise affect collective behavior when socially or culturally distinct perspectives are represented within the agent population. Multi-agent cultural debate shows that explicitly representing different cultural perspectives can reduce cultural bias, while excessive deliberation can weaken these differences by smoothing away useful cultural distinctions [139].

Ethical requirements face a related system-level problem because a constraint stated at the beginning of an execution may not retain the same force throughout a multi-agent trajectory. Li et al. [83] describe *constraint drift*, in which requirements can be weakened, omitted, or transformed as information passes through memory, delegation, communication, tools, and other stages of execution. Their analysis shows why an acceptable final response does not necessarily imply that the process leading to it remained compliant. Ethical behavior in LLM-MAS should therefore be considered across the trajectory, including how requirements are preserved and acted upon during interaction, rather than inferred solely from the terminal output.

Together, this literature suggests that fairness and ethical behavior in LLM-MAS must be considered across individual agents, their interactions, and the collective outcomes that emerge from them. Rather than proposing an exhaustive decomposition of fairness or ethics, we organize the literature reviewed here around three recurring concerns. *Equity* concerns whether benefits, burdens, opportunities, resources, services, and treatment are distributed without unjustified disadvantage. *Diversity* concerns whether relevant social, cultural, and stakeholder perspectives are represented and remain meaningfully present in collective reasoning and decision-making. Here, diversity refers to representation and inclusion in a fairness-related sense rather than heterogeneity introduced solely to improve model performance. *Ethical norm adherence* concerns whether agent actions and interactions continue to satisfy specified ethical, social, institutional, professional, or legal norms and constraints throughout execution. These three concerns provide the basis for the discussion that follows.

5.1 Equity

Equity in AI and Machine Learning. *Equity* is closely related to fairness but should not be reduced to identical treatment or identical outcomes. Machine learning research has formalized fairness through a range of individual and group-based criteria that examine whether people receive comparable treatment, opportunities, or predictive quality [97]. Equity places greater emphasis on whether such decisions create or reinforce unjustified disadvantage and recognizes that avoiding disadvantage does not necessarily require treating all individuals or groups identically. This distinction is particularly important when AI systems influence access to consequential opportunities, resources, or services. Fairlearn similarly frames fairness in terms of harms experienced by affected groups, including unequal allocation of opportunities and differences in the quality of service delivered by a system [155]. In this survey, we therefore use equity to refer to whether an AI system produces or reinforces unjustified disparities in benefits, burdens, opportunities, resources, services, or consequential treatment across relevant individuals or groups. We reserve questions about the representation and preservation of different perspectives for Diversity and questions about appropriate conduct during interaction for Ethical Norm Adherence.

Equity in Multi-Agent Systems. In multi-agent systems, the question of equity extends from the treatment of individuals by a model to the distribution of benefits and burdens among interacting participants. Optimizing aggregate system performance can produce efficient collective outcomes while leaving particular agents systematically disadvantaged. Early work on multi-agent sequential decision making addresses this tension by combining protection of the least advantaged participant with overall system utility [180]. Related work shows that fairness objectives can be incorporated directly into decentralized agent policies, allowing agents to coordinate resource use while balancing individual utilities with collective efficiency [68]. These studies establish that equity in a multi-agent setting concerns not only the quality of the final collective outcome, but also how the gains, costs, and resources generated through coordination are distributed among participants.

Equity in LLM-Based Multi-Agent Systems. LLM-based systems introduce additional sources of inequity because judgments can depend on social identity or contextual information that should not determine the outcome. Counterfactual evaluation shows that otherwise comparable behavior can receive different assessments when demographic, historical, or behavioral attributes are changed while substantive task information is preserved [95]. In LLM-MAS, the same problem extends to settings in which agents judge, critique, rank, or respond to one another. MALIBU demonstrates that multi-agent judging systems can assign different evaluations to comparable responses when demographic identities are attached to them [99]. Demographic personas can further change how readily an agent's position is accepted,

how strongly agents maintain their positions, and how conformity develops over repeated interaction [79]. Equity in LLM-MAS therefore concerns both differential treatment in final outcomes and whether socially irrelevant identity cues change how contributions are evaluated or how much influence participants receive during collective reasoning.

Equity at the system level cannot generally be inferred from the behavior of individual agents. Collaboration can either reduce or amplify group disparities, and some multi-agent configurations produce substantially greater disparities than those observed for their constituent agents [92]. Communication can itself change the bias expressed by the group, with stereotypical responses emerging during interaction and spreading across agents over repeated rounds [104]. Conformity can similarly transform relatively modest individual biases into strongly biased collective consensus, while agent heterogeneity and interaction conditions affect whether this transition occurs [106]. These findings show that inequity can emerge from the interaction process rather than merely being inherited from the underlying models.

System-level equity also includes the distribution of shared resources and collective gains. Multi-agent resource management and stakeholder negotiation show that successful collective action can coexist with unequal distributions of benefits and burdens [49, 113]. Interaction structure further shapes these outcomes. Agent ordering, communication connectivity, and aggregation procedures affect which contributions receive greater influence and can alter fairness-related outcomes even when the participating models remain unchanged [9]. Broader work on fairness in agentic AI consequently treats fairness as a dynamic system property shaped by interactions, incentives, and correction mechanisms rather than as a constraint on isolated decisions [121]. Equity evaluation should therefore examine both outcome distributions and the interaction mechanisms that produce them.

Evaluating and Mitigating Inequity. Approaches to equity intervene at different stages of the decision process. Collaborative fairness auditing improves the estimation of group disparities through coordinated auditing agents [36], while aggregation methods such as REQUAL-LM explicitly incorporate equity into the selection of collective outputs [43]. In LLM-MAS, mitigation can also be integrated directly into the workflow. MOMA modifies social-group information before task execution to reduce social bias while limiting loss in task performance [168], and broader agentic fairness frameworks introduce fairness constraints, bias-sensitive incentives, and adaptive correction within the coordination process [121]. These approaches illustrate that inequity can be addressed through several intervention points, including auditing, input transformation, aggregation, incentive design, and changes to the interaction process itself. The appropriate intervention depends on whether the disparity originates in the underlying model, the treatment of participants during interaction, the aggregation of their contributions, or the distribution of the resulting benefits and burdens. Additional empirical results and implementation details for equity evaluation and mitigation are provided in Appendix F.10.

5.2 Diversity

Diversity in AI and Machine Learning. *Diversity* in responsible AI concerns whether the people, experiences, values, and perspectives relevant to a system's use are adequately represented when the system is designed, evaluated, and used. This concern is closely connected to fairness because aggregate evaluation can obscure differences in how a system serves particular groups, while a narrow set of assumptions or perspectives can leave the needs of affected populations unrepresented [97, 155]. Diversity in this sense is not limited to the numerical presence of different demographic groups. It also concerns whether socially, culturally, or institutionally relevant perspectives are recognized when defining desirable system behavior and evaluating its consequences. In this survey, we therefore use diversity to refer to the representation and meaningful inclusion of relevant social, cultural, stakeholder, and value perspectives.

This distinguishes fairness-related diversity from technical heterogeneity introduced solely to improve model accuracy, efficiency, or robustness.

Diversity in Multi-Agent Systems. The role of diversity becomes more explicit in multi-agent systems because collective decisions can be produced by participants with different information, interests, preferences, roles, or objectives. Such heterogeneity creates an opportunity to incorporate perspectives that would otherwise be absent from a single decision maker, but the presence of different agents does not by itself ensure meaningful inclusion. A collective process may still privilege particular participants, suppress competing positions, or aggregate away differences before they can affect the outcome. Diversity in a fairness-related sense therefore concerns both who or what is represented in the agent population and whether these differences remain consequential during coordination. This distinction is particularly important for human-centered multi-agent systems, where the relevant perspectives should be connected to the people and stakeholders affected by the resulting decisions rather than chosen only for their usefulness to task performance. Recent reviews of multi-agent AI fairness similarly emphasize the need to identify explicit fairness beneficiaries and affected human stakeholders rather than evaluating fairness only within populations of artificial agents [4].

Diversity in LLM-Based Multi-Agent Systems. LLM-based multi-agent systems make diverse agent characteristics relatively easy to instantiate because agents can be assigned different social identities, cultural backgrounds, stakeholder roles, personalities, or value profiles through natural language. LLM-MAS research also uses diversity to describe technical heterogeneity. X-MAS combines different language models across agent roles to exploit complementary capabilities, while SMoA uses role-specific prompting to encourage diverse reasoning patterns [76, 174]. Although such heterogeneity can improve performance and may affect fairness-related outcomes, we distinguish it from the fairness-related diversity considered here, which concerns the representation of social, cultural, stakeholder, or value perspectives.

Differences in socially relevant agent characteristics can materially shape collective behavior. FAIRGAME shows that strategic behavior varies systematically across languages and personality configurations in repeated multi-agent games [15]. Such variation indicates that socially and linguistically framed differences can influence interaction, but their presence alone does not ensure meaningful representation or inclusion. Multi-Agent Cultural Debate provides a more direct example of fairness related diversity by assigning agents culturally grounded perspectives and allowing them to deliberate before producing a shared answer [139]. Its ablations show that removing the cultural personas substantially reduces the fairness benefit, suggesting that the represented perspectives themselves contribute to the improvement rather than multi-agent interaction alone. Similarly, MAJ-EVAL derives evaluator agents from different stakeholder groups and their domain-specific priorities, allowing evaluation to incorporate multiple legitimate perspectives rather than relying on a single uniform judge [21]. These approaches show how LLM-MAS can incorporate both cultural representation and stakeholder inclusion into collective reasoning and evaluation.

Fairness-related diversity is not limited to demographic, cultural, or stakeholder identities. Differences in underlying values can also shape the collective behavior that emerges from an LLM-agent community. Huang et al. [64] show that communities composed of agents with different value profiles develop different interaction structures, deliberative patterns, and collective governance norms than homogeneous groups. Their findings suggest that value diversity can broaden collective behavior while also creating coordination costs when heterogeneity becomes excessive. This extends the role of diversity in LLM-MAS to settings where agents represent people or institutions whose disagreements arise not only from social or cultural differences, but also from competing priorities and normative commitments.

Preserving Diversity Through Interaction. Representation at initialization does not guarantee that diversity will survive the interaction process. Communication and aggregation can cause agents to converge, allowing initially distinct perspectives to lose their influence over the collective outcome. Okawa [106] show that homogeneous agent populations are particularly susceptible to conformity-driven biased consensus, while greater heterogeneity can reduce this tendency. Cultural diversity can similarly be weakened by excessive deliberation. In Multi-Agent Cultural Debate, additional debate rounds eventually reduce performance, which the authors associate with excessive smoothing of culturally distinctive information [139]. At the same time, diversity is not inherently beneficial. Social group differences can interact with communication in ways that reinforce stereotypes and ingroup preferences rather than producing more balanced reasoning [104]. Diversity in LLM-MAS should therefore be evaluated not only by whether different perspectives are present, but also by whether the interaction process allows them to remain meaningfully represented without producing exclusion, stereotyping, or premature homogenization.

5.3 Ethical Norm Adherence

Ethical Norm Adherence in AI and Machine Learning. Ethical norm adherence concerns whether an AI system behaves consistently with the ethical principles, social and professional norms, institutional policies, legal requirements, and safety constraints relevant to its use. We distinguish this concept from the broader notion of value alignment. Value alignment may concern compatibility with human preferences, goals, intentions, or desired behavior more generally, whereas ethical norm adherence focuses on normatively relevant expectations regarding what a system should or should not do. These expectations may concern non-harm, privacy, fairness, respectful conduct, professional responsibility, or compliance with institutional and legal requirements. In this survey, we therefore use ethical norm adherence to describe whether such expectations remain reflected in system behavior in the situations in which they apply.

Early approaches to aligning language models with desired behavior have largely relied on supervised instruction tuning, reinforcement learning from human feedback, and related preference-based methods [58]. Although these approaches can promote safer and more desirable behavior, they do not establish whether specific ethical, professional, or institutional norms will be satisfied across different contexts. Ethical norm adherence therefore requires evaluating system behavior against the particular normative expectations relevant to its deployment.

Ethical Norm Adherence in Multi-Agent Systems. Norms become especially important in multi-agent systems because autonomous participants operate within shared environments and their actions can affect other participants and external stakeholders. Normative approaches to multi-agent systems consequently consider how obligations, prohibitions, and other expected behaviors are represented, monitored, and enforced. More recently, He et al. [56] examine whether LLMs can support this process by detecting norm violations in multi-agent interactions. Their results illustrate both the potential of automated normative monitoring and the difficulty of reliably identifying all violations. Ethical norm adherence at the multi-agent level therefore concerns not only whether individual agents satisfy prescribed norms, but also whether the collective process can preserve, monitor, and respond to those norms during interaction.

Monitoring may further be coupled with active enforcement. Enforcement Agents introduce dedicated supervisory agents that observe runtime behavior, identify policy violations or anomalous actions, intervene against misbehaving agents, and report corrective actions [138]. Although the framework is primarily motivated by accountability and resilience, it illustrates an important aspect of normative multi-agent systems. Norms require mechanisms that can detect and respond to violations while collective activity is still underway rather than relying only on post-hoc inspection.

Ethical Norm Adherence in LLM-Based Multi-Agent Systems. LLM-MAS make norm adherence more difficult because requirements expressed in natural language must remain effective as agents communicate, delegate tasks, update memory, use tools, and influence one another. Individually aligned agents therefore do not necessarily produce a norm-adherent collective system. A requirement available to one agent may be omitted during delegation, weakened through summarization, interpreted differently by another agent, or subordinated to task completion. Recent work also emphasizes the importance of making normative objectives and affected stakeholders explicit rather than assuming that broad ethical prompts provide an adequate specification of responsible behavior [4]. Ethical norm adherence in LLM-MAS should therefore be evaluated as a system-level property that persists throughout collective execution.

Norm Compliance and Constraint Preservation. A central challenge is that successful task completion does not imply that normative requirements were preserved during execution. MAC-Bench evaluates whether LLM-based multi-agent systems satisfy legal, security, privacy, ethical, and organizational requirements while completing assigned tasks [192]. Its results show substantial separation between task success and compliance and further indicate that adherence depends on the coordination architecture. For GPT-4o under high operational pressure, the hierarchical AutoGen configuration achieves 96.8% task success but only 38.5% compliance, while ChatDev achieves 89.1% task success and 70.3% compliance. Successful collective execution therefore does not establish that the norms governing the task were preserved throughout the process.

A related failure occurs when a norm is initially available but gradually loses its operational force. Li et al. [83] describe this phenomenon as *constraint drift*, in which safety critical constraints can be lost, distorted, or weakened through memory, authority delegation, information flow, auditing, or optimization. Their results show that inspection of the final output can conceal violations that occurred internally. Their Constraint State Governance framework instead maintains constraints as explicit execution state and checks their continued validity before consequential actions. EnviSmart follows a related principle by externalizing governance rules, authorization constraints, and safety boundaries as persistent artifacts and treating agent handoffs as explicit trust boundaries [53]. These approaches illustrate how normative requirements can be preserved across execution rather than repeatedly reconstructed from natural language context.

Moral, Professional, and Interactional Norms. Ethical norm adherence is broader than compliance with formal rules and may also concern socially grounded moral expectations and professional standards. FairMindSim evaluates behavior in ethical dilemmas involving unfair resource allocation and examines whether behavior remains consistent with expectations concerning fairness and justice [75]. In high stakes domains, normative criteria can instead be grounded in professional standards. MedSentry evaluates medical LLM-MAS using criteria derived from the AMA Principles of Medical Ethics and considers harms including unsafe medical advice, privacy violations, discriminatory care, and risks to vulnerable populations [23]. Its results further show that safety outcomes vary with interaction topology and that unsafe behavior can propagate through the collective process. These examples extend norm adherence beyond generic alignment toward identifiable moral and professional expectations against which system behavior can be evaluated.

Norms can also govern the interaction process itself. Interactional Fairness evaluates communication through expectations of respectful and dignified treatment and evaluates explanations through their clarity and adequacy [11]. Its resource allocation experiments show that respectful communication and clear justification can affect whether proposals are accepted even when the underlying allocation remains unchanged. Normative principles can likewise be incorporated directly into collective reasoning through specialized ethics roles or explicit principles governing agent

behavior [35, 113]. Ethical norm adherence therefore concerns not only whether the final outcome satisfies a rule, but also whether the interactions and decisions producing that outcome remain consistent with relevant expectations.

Verifying Ethical Norm Adherence. Because violations may occur internally even when the final outcome appears acceptable, evaluating norm adherence requires evidence from the execution trajectory. HarnessAudit evaluates agent systems against specified tool, resource, permission, and information flow boundaries throughout execution [89]. Its results show that multi-agent systems can complete tasks while violating intermediate constraints and that collaboration enlarges the surface over which such violations can occur. Although HarnessAudit primarily contributes to Auditability, its trajectory evidence provides a means of determining whether specified constraints were actually respected during collective execution.

These studies characterize ethical norm adherence in LLM-MAS as a system and trajectory property rather than a property of individual models or final outputs alone. Relevant norms may be moral, social, professional, institutional, or legal, and their effectiveness depends on whether they remain operative as agents communicate, delegate, reason, and act. Responsible LLM-MAS therefore require mechanisms that make applicable norms explicit, preserve them across interaction, detect and respond to violations, and provide evidence that they were respected throughout execution. Additional implementation details and quantitative results are provided in Appendix F.11.

6 Relationships

The four dimensions in our taxonomy are conceptually distinct but closely interrelated. *Reliability* concerns whether an LLM-based multi-agent system performs accurately and consistently and can maintain or restore its function under perturbations and failures. *Transparency and Understandability* concern whether system behavior, execution processes and intermediate reasoning artifacts, uncertainty, limitations, and experimental conditions can be inspected, understood, and reproduced. *Accountability* concerns whether responsibility for actions, contributions, influence, and failures can be established and supported by evidence for review, remediation, or recourse. *Fairness and Ethics* concern whether collective behavior avoids unjustified disadvantage, incorporates relevant social and value perspectives, and remains consistent with applicable ethical norms. Clarified the relationships among the four dimensions and modeled them as overlapping rather than hierarchically nested. Although these dimensions address different properties of responsible LLM-based multi-agent systems, they frequently interact in both system design and evaluation. Transparency can provide evidence needed for accountability, accountability mechanisms can support the diagnosis and repair of reliability failures, and reliable operation can help fairness or ethical constraints remain effective under changing conditions. At the same time, these relationships do not imply containment. A system may be reliable without being transparent, transparent without being fair, or traceable without providing a sufficient basis for assigning responsibility. We therefore model the dimensions as overlapping rather than nested and use their intersections to characterize work that substantively addresses more than one dimension.

6.1 *Transparency and Understandability ∩ Reliability*

Explainability and Reliability. Explainability can support reliability when intermediate outputs and feedback are made visible and used to identify and correct errors before they propagate through the system. Layered CoT introduces verification between reasoning stages to detect inconsistencies before they affect subsequent steps [125], while reflective multi-agent collaboration uses reviewer feedback and revision to correct deficient intermediate outputs [12]. In both cases, visibility supports reliability only when coupled with verification or correction.

Uncertainty and Reliability. Uncertainty information can indicate when collective outputs are unstable and trigger verification, deferral, escalation, or alternative aggregation. Chen et al. [25] quantify uncertainty from variation across agents, reasoning steps, repeated executions, and communication structures, while related reliability frameworks use confidence estimates to support risk-aware decision policies [74, 111]. Importantly, uncertainty can itself degrade during coordination. Emde et al. [46] show that uncertainty associated with an upstream response may be diminished when communicated to downstream agents, causing uncertain information to appear more reliable as it propagates. Preserving uncertainty across agent handoffs is therefore relevant to maintaining reliable collective behavior.

Reproducibility and Reliability. Reproducibility provides the experimental basis for determining whether observed reliability is stable rather than an artifact of stochastic variation. Reporting model configurations, prompts, interaction protocols, and results of repeated runs allows performance differences to be assessed under comparable conditions. Kaliyev and Maryanskyy [70] show that equivalent multi-agent configurations can exhibit substantial run-to-run variation and propose a paired noise floor protocol for determining whether reported coordination gains exceed this variability. Therefore, reproducibility helps validate reliability claims without guaranteeing reliable behavior.

The relationship between the two dimensions is consequently supportive rather than implicational. A transparent system may expose its intermediate outputs, uncertainty, and failure modes while still producing inaccurate or brittle results. For instance, natural language debate provides an inspectable interaction record but remains susceptible to persuasive adversarial agents that can steer the collective decision toward an incorrect outcome [6]. Conversely, reliability does not require the underlying coordination mechanism to be human interpretable. Communication through internal representations can improve task performance by exchanging latent activations rather than natural language messages [120], while state delta communication can preserve information relevant to decisions without producing an interaction trace that is directly interpretable to humans [140]. Such systems may improve accuracy or robustness while offering limited explainability of the exchanged representations.

Transparency and Understandability support Reliability by making failures, uncertainty, limitations, and experimental variability more observable and hence more amenable to verification or correction. However, reliability ultimately depends on whether the system continues to perform acceptably under repeated and adverse conditions. Therefore, the intersection captures systems in which transparency information is substantively used to evaluate, diagnose, or maintain reliable behavior rather than serving as evidence of reliability by itself.

6.2 *Transparency and Understandability ∩ Fairness and Ethics*

Transparency and Understandability support Fairness and Ethics by making the evidence, assumptions, and interaction processes underlying fairness and ethical judgments open to examination. This is particularly important in LLM-MAS, where disparities, exclusion of perspectives, or norm violations may emerge through communication, aggregation, and delegation rather than being apparent from the final outcome alone.

Can fairness and ethics claims stand without Transparency and Understandability? Fairness and ethics do not logically imply transparency, but empirically substantiating such claims generally relies on observable evidence about the behavior being assessed. Detecting subgroup disparities, determining whether different perspectives remain influential during interaction, or evaluating adherence to an ethical norm all depend on evidence that can be inspected and interpreted. Transparency and Understandability therefore strengthen fairness and ethics claims by making their empirical and normative basis more open to scrutiny. As noted in [3], ["BIAS IS TIGHTLY ASSOCIATED WITH THE CONCEPTS

of transparency as well as fairness in society."] In LLM-MAS, this connection is especially important because interaction itself may introduce disparities or violations that are not evident from isolated agent outputs.

For *equity*, transparency can expose how disparities are identified, measured, and mitigated. For example, Collaborative fairness auditing coordinates multiple auditors to estimate demographic parity gaps under limited query budgets while making the sampling and evaluation procedure explicit [36]. MOMA similarly exposes its intervention process through explicit masking and counterfactual balancing of social group information [168]. Interactional Fairness provides an even more direct connection to understandability by treating respectful communication and clear, adequate explanations as observable fairness properties [11].

For *diversity*, visibility into the interaction process helps determine whether represented perspectives remain consequential rather than merely being present at initialization. Multi-Agent Cultural Debate shows that culturally grounded perspectives can improve fairness, while excessive deliberation can weaken their distinctive contribution through convergence [139]. MAJ-EVAL similarly makes stakeholder perspectives and their evaluation criteria explicit, allowing the contribution of different stakeholder priorities to be examined systematically [21].

For *Ethical Norm Adherence*, transparency makes it possible to examine whether specified requirements remain operative throughout execution. Li et al. [83] maintains critical safety constraints as explicit execution state so that their preservation across memory, delegation, and tool use can be inspected. More broadly, explicitly specifying normative objectives, intended beneficiaries, affected stakeholders, and evaluation criteria reduces ambiguity about what behavior is being judged as fair or ethically appropriate [4].

Reproducibility provides a complementary connection by allowing fairness and ethical claims to be tested across repeated runs and system configurations. FAIRGAME uses controlled and reproducible variations in language, personality, and strategic conditions to expose behavioral differences [15], while reproducibility studies of stakeholder negotiation examine whether conclusions about cooperation and inequality persist under alternative models and prompting conditions [49]. Such evaluation is important because fairness outcomes can depend substantially on agent composition, interaction structure, and communication conditions.

The relationship between the two dimensions is consequently supportive rather than implicational. Transparent interactions can still produce biased or ethically undesirable outcomes, as communication may amplify stereotypes or drive agents toward biased consensus [104, 106]. Conversely, a system may reduce a measured disparity without making its underlying coordination process fully understandable. Transparency and Understandability therefore make fairness and ethical behavior more inspectable, interpretable, and verifiable, but should not themselves be treated as evidence that fairness or ethical conduct has been achieved.

6.3 *Transparency and Understandability ∩ Accountability*

Transparency and Understandability provide much of the evidentiary basis on which Accountability operates. Information about agent actions, messages, tool use, memory, intermediate artifacts, and system state can make a multi-agent trajectory inspectable and understandable. Accountability uses this evidence for a further purpose by determining which agents or interactions contributed to an outcome, whether relevant responsibilities were satisfied, and whether the resulting behavior can support independent review [159, 197]. The relationship is therefore close but not equivalent.

Trace Evidence and Attribution. Attribution depends strongly on what information about an execution is available. Evaluating only the final output may reveal that a system failed without showing where the decisive error originated or how it propagated through later agents. TraceElephant demonstrates this directly, finding that access to complete

inputs, intermediate outputs, tool interactions, and environmental context improves agent-level attribution accuracy by 22% and step-level accuracy by 76% over output traces [24]. Related failure attribution methods use such evidence to identify responsible agents and decisive steps across long interaction histories [10, 188, 196]. Transparent execution evidence supports accountability by making the actions and dependencies needed for attribution available for analysis.

Provenance, Reconstruction, and Audit. Accountability also depends on understanding how information moves through the system. Chronological logs show what occurred, while provenance representations additionally connect later actions and outputs to the agents, messages, tools, memory items, and data that supported them. PROV-AGENT formalizes these dependencies through standardized provenance relations that support both backward tracing from an outcome to its sources and forward tracing from an agent action to its downstream consequences [133]. AgentTrace similarly connects cognitive, operational, and contextual events through structured trace and span relationships [5]. Such provenance makes system behavior more reconstructable and provides evidence that can subsequently be examined against policies, permissions, or other governance requirements. Therefore, Transparency contributes to auditability when visible records are sufficiently structured to support reconstruction and independent review.

Visibility and Responsibility. Visibility alone does not establish responsibility. A trajectory may record every message and action while leaving open whether responsibility belongs to the agent that introduced an error, another agent that amplified it, an orchestrator that routed it, or a verifier that failed to intervene. This ambiguity is visible in MP-Bench, where only 16.2% of annotated failure steps were selected by all three expert annotators [65]. More generally, complete observability can reveal how a collective failure developed without determining how responsibility should be distributed across interacting agents [32, 159]. Therefore, Accountability requires additional causal and normative interpretation of roles, contributions, obligations, and failures rather than visibility alone. Transparency and Understandability make the evidence needed for attribution, reconstruction, and audit more accessible and interpretable, while Accountability determines how that evidence should be used to assign responsibility and support review or recourse. A system can therefore be highly observable without providing a sufficient basis for accountability.

6.4 *Reliability ∩ Fairness and Ethics*

This overlap captures systems that pursue technical dependability and fairness and ethical objectives together. In LLM-MAS, the two can come into tension. Mechanisms that improve accuracy, consensus, or robustness may also stabilize or amplify biased behavior, while fairness interventions may lose their benefits if they are not robust to changes in agents, prompts, topology, or operating conditions. REQUAL-LM makes this coupling explicit by using aggregation to improve decision reliability while reducing inequity across groups [43]. MOMA similarly treats bias reduction and task performance as a multi-objective tradeoff, substantially reducing social bias while controlling the associated loss [168].

The same issue arises for *Ethical Norm Adherence*. A system may perform its task reliably while failing to preserve the constraints governing how that task should be carried out. Constraint Drift shows that requirements can be weakened or lost as they pass through delegation, memory, communication, and tool use [83], while MAC-Bench demonstrates that high task success can coexist with substantially lower compliance [192]. MedSentry further shows that safety can change with interaction topology and deteriorate when malicious agents enter the system [23].

The key question at this intersection is therefore not simply whether a system is reliable or fair in isolation, but whether fairness and ethical constraints remain effective under relevant variations and disturbances. Reliability gains should not come at the expense of whom the system disadvantages or which norms it violates, and fairness improvements should not disappear as soon as the multi-agent setting changes.

6.5 *Reliability ∩ Accountability*

Reliability and Accountability intersect when evidence about who or what contributed to an outcome is used to diagnose failures and restore dependable operation. In LLM-MAS, a system may detect that performance has degraded without knowing which agent, interaction, or execution step should be changed. Failure attribution narrows this gap by identifying the components responsible for consequential errors. Zhang et al. [188] formalize this through the decisive agent-step pair associated with a failed trajectory, while TraceElephant shows that richer execution traces substantially improve the accuracy of such localization [24]. Accountability mechanisms therefore support reliability when they turn system failures into actionable diagnoses rather than merely post-hoc records.

The connection becomes stronger when diagnosis is coupled directly to repair. MASC identifies anomalous agent outputs before they propagate and routes them to a correction agent [129], while HiveMind uses agent-level contribution estimates to identify weak components and refine their prompts during operation [165]. Related provenance and audit mechanisms can support rerouting, isolation, rollback, or escalation by showing which agent, message, or action introduced the failure. These systems move toward the cycle of attribution, verification, and repair identified as a major open challenge in current LLM-MAS research [115].

The two dimensions nevertheless capture different properties. A system can be highly reliable while providing little basis for determining which agents were responsible for its decisions, and a fully traceable and attributable system can repeatedly produce incorrect or brittle outcomes. Their intersection is most meaningful when accountability evidence is used operationally to detect, localize, and correct the failures that threaten continued reliable behavior.

6.6 *Accountability ∩ Fairness and Ethics*

Accountability intersects with Fairness and Ethics when disparities, harmful interactions, or norm violations must be connected to the agents and decisions that produced them. Detecting an inequitable outcome is not always sufficient for remediation because multi-agent behavior may emerge from several agents, communication rounds, or aggregation decisions. Collaborative fairness auditing illustrates this connection by coordinating auditors to estimate group disparities while producing evidence that can support systematic review of the resulting fairness claims [36]. Interactional Fairness similarly evaluates fairness at the level of individual exchanges, making it possible to identify particular interactions in which respectful treatment or adequate explanation breaks down [11].

The connection is particularly important for *Ethical Norm Adherence*, where accountability mechanisms can determine when and where a requirement was lost or violated. Constraint State Governance records active constraints, delegated authority, proposed actions, and validation decisions so that constraint failures can be traced across execution [83]. HarnessAudit likewise evaluates complete trajectories against tool, resource, permission, and information-flow boundaries, showing that successful task completion can coexist with violations in how the task was carried out [89]. Such mechanisms make ethical failures not only detectable but also reviewable at the level of the responsible actions and interactions.

Nevertheless, the boundary between the two dimensions is important. Accountability can establish who acted, what they contributed, and which obligations were violated without ensuring that the resulting system is fair or ethical. Conversely, a system may satisfy an equity metric or produce an ethically acceptable outcome while providing little basis for determining who was responsible for that outcome or how recourse should proceed when failures occur. Their intersection therefore concerns whether fairness and ethical harms can be traced to responsible parts of the multi-agent process and supported by evidence sufficient for review, remediation, or recourse.

7 Future Work

As LLM based multi-agent systems move into safety critical and human facing settings, stronger guarantees are needed for coordination, recovery, responsibility, transparency, and fairness. Despite recent progress, important gaps remain across all four dimensions of our taxonomy.

Reliability: Reliability remains largely empirical, with topology, role assignment, and coordination structure often selected through task specific experimentation rather than principled design. Future work should develop methods that adapt communication structure, agent roles, and aggregation to task characteristics and competing objectives such as accuracy, cost, latency, robustness, and recoverability [81, 117, 146, 162]. Recoverability also remains underdeveloped, as many systems still rely on retries or restarts rather than explicit recovery protocols. Promising directions include coordinated checkpointing, rollback, rerouting, team repair, and standardized measures of recovery performance.

Accountability: A major open problem is reliable contribution and influence attribution under communication and inter-agent dependence. Existing approaches based on outcome signals, ablations, or LLM judges can be costly and distorted by persuasion, memory, and information that persists after an agent is removed [12, 33, 42]. Future work should develop scalable causal attribution methods that explicitly model communication dependencies and distinguish original contribution from propagated influence.

Transparency and Understandability: Emerging communication mechanisms create a tradeoff between performance and inspectability. Communication through embeddings, activations, or other latent representations may improve coordination [26, 112, 120, 140, 194], but can make interaction harder to interpret, reconstruct, and audit. Future work should therefore seek communication mechanisms that retain these efficiency gains while preserving sufficient evidence for explanation, provenance, and review.

Fairness and Ethics: Fairness in LLM based multi-agent systems remains insufficiently characterized as a dynamic property of interaction. Inequity may arise through unequal participation, resource access, persuasive influence, aggregation, or repeated coordination rather than only in final outcomes. Future work should develop process level measures that track how influence, evidence, resources, and outcomes evolve across interaction rounds. A related challenge is determining when diversity and heterogeneity improve collective behavior. Differences in model capability, social or cultural perspective, role specialization, and knowledge may improve reasoning but can also increase coordination costs or amplify disagreement. Future research should develop principled measures and predictive guidelines for when heterogeneous agent populations improve performance and fairness under practical constraints [111, 174].

8 Conclusion

LLM-based multi-agent systems extend the capabilities of individual models through communication, coordination, and collective reasoning, but these same interactions introduce new sources of failure and responsibility. This survey organized the growing literature around four dimensions: Reliability, Transparency and Understandability, Accountability, and Fairness & Ethics. Across these dimensions, a common finding is that responsible LLM-MAS cannot be assessed solely through the properties of individual agents or final outputs. Communication, delegation, aggregation, memory, tool use, and interaction structure can change how errors, uncertainty, influence, and normative constraints develop across the system.

The relationships among these dimensions are therefore as important as the dimensions themselves. Transparency and understandable execution evidence can support reliability, attribution, and review, but visibility alone does not

establish accountability. Reliability mechanisms can stabilize system behavior without ensuring that the resulting process is fair or ethically acceptable, while fairness and ethical constraints must remain effective as agents, interactions, and operating conditions change. Accountability similarly requires more than complete logging, particularly when outcomes emerge from distributed causal influence rather than a single identifiable action. These dependencies show that responsibility in LLM-MAS is a system-level property shaped by both agent behavior and the structures through which agents interact.

Responsible LLM-MAS should therefore combine reliable execution with sufficient evidence, verification, recovery, attribution, and safeguards for fairness and ethical conduct. Provenance and uncertainty information should be connected to mechanisms for verification and intervention; recovery should be supported through checkpoints, monitoring, and reconfiguration; and contribution and influence should be examined across the interaction process rather than inferred only from final outcomes. The taxonomy and synthesis presented in this survey provide a common foundation for analyzing these requirements and for designing LLM-MAS that remain inspectable, accountable, recoverable, and fair as their complexity increases.

References

[1] [n. d.]. *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 on Artificial Intelligence and amending Regulations (EU) 2021/0106 and (EU) 2021/695 (Artificial Intelligence Act).* https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=OJ%3AL_202401689

[2] Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. 2023. Llm-coordination: evaluating and analyzing multi-agent coordination abilities in large language models. *arXiv preprint arXiv:2310.03903* (2023).

[3] NIST AI. 2023. Artificial intelligence risk management framework (AI RMF 1.0). *URL: https://nvlpubs.nist.gov/nistpubs/ai/nist.ai* (2023), 100–1.

[4] Simeon Allmendinger, Luca Deck, and Lucas Mueller. 2026. Where are the Humans? A Scoping Review of Fairness in Multi-agent AI Systems. *arXiv preprint arXiv:2604.15078* (2026).

[5] Adam AlSayyad, Kelvin Yuxiang Huang, and Richik Pal. 2026. AgentTrace: A Structured Logging Framework for Agent System Observability. *arXiv preprint arXiv:2602.10133* (2026).

[6] Alfonso Amayuelas, Xianjun Yang, Antonis Antoniades, Wenyue Hua, Liangming Pan, and William Yang Wang. 2024. MultiAgent Collaboration Attack: Investigating Adversarial Attacks in Large Language Model Collaborations via Debate. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (Eds.). Association for Computational Linguistics, Miami, Florida, USA, 6929–6948. doi:10.18653/v1/2024.findings-emnlp.407

[7] Maryam Amirizaniani, Jihan Yao, Adrian Lavergne, Elizabeth Snell Okada, Aman Chadha, Tanya Roosta, and Chirag Shah. 2024. LLMAuditor: A Framework for Auditing Large Language Models Using Human-in-the-Loop. *arXiv preprint arXiv:2402.09346* (2024).

[8] Ruicheng Ao, Siyang Gao, and David Simchi-Levi. 2026. On the Reliability Limits of LLM-Based Multi-Agent Planning. *arXiv preprint arXiv:2603.26993* (2026).

[9] Tanav Singh Bajaj, Nikhil Singh, Karan Anand, and Eishkaran Singh. 2026. Position: Safety and Fairness in Agentic AI Depend on Interaction Topology, Not on Model Scale or Alignment. *arXiv preprint arXiv:2605.01147* (2026).

[10] Shraddha Barke, Arnav Goyal, Alind Khare, Avaljot Singh, Suman Nath, and Chetan Bansal. 2026. AgentRx: Diagnosing AI Agent Failures from Execution Trajectories. *arXiv preprint arXiv:2602.02475* (2026).

[11] Ruta Binkyte. 2025. Interactional Fairness in LLM Multi-Agent Systems: An Evaluation Framework. *arXiv preprint arXiv:2505.12001* (2025).

[12] Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. 2024. Reflective Multi-Agent Collaboration based on Large Language Models. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 138595–138631. https://proceedings.neurips.cc/paper_files/paper/2024/file/fa54b0edce5eef0bb07654e8ee800cb4-Paper-Conference.pdf

[13] Mark Bovens. 2007. Analysing and Assessing Accountability: A Conceptual Framework. *European Law Journal* 13, 4 (2007), 447–468. doi:10.1111/j.1468-0386.2007.00378.x

[14] Houssem Ben Braiek and Foutse Khomh. 2025. Machine learning robustness: A primer. In *Trustworthy AI in medical imaging*. Elsevier, 37–71.

[15] Alessio Buscemi, Daniele Proverbio, Alessandro Di Stefano, The Anh Han, German Castignani, and Pietro Liò. 2025. Fairgame: a framework for ai agents bias recognition using game theory. *arXiv preprint arXiv:2504.14325* (2025).

[16] Yuandao Cai, Wensheng Tang, Cheng Wen, and Shengchao Qin. 2026. Ghost in the agent: Redefining information flow tracking for llm agents. *arXiv preprint arXiv:2604.23374* (2026).

[17] Mert Cemri, Melissa Z Pan, Shuyi Yang, Lakshya A Agrawal, Bhavya Chopra, Rishabh Tiwari, Kurt Keutzer, Aditya Parameswaran, Dan Klein, Kannan Ramchandran, et al. 2025. Why do multi-agent llm systems fail? *arXiv preprint arXiv:2503.13657* (2025).

[18] Alan Chan, Carson Ezell, Max Kaufmann, Kevin Wei, Lewis Hammond, Herbie Bradley, Emma Bluemke, Nitarshan Rajkumar, David Krueger, Noam Kolt, et al. 2024. Visibility into AI agents. In *Proceedings of the 2024 ACM conference on fairness, accountability, and transparency*. 958–973.

[19] Bei Chen, Gaolei Li, Xi Lin, Zheng Wang, and Jianhua Li. 2024. BlockAgents: Towards Byzantine-Robust LLM-Based Multi-Agent Coordination via Blockchain *(ACM-TURC '24)*. Association for Computing Machinery, New York, NY, USA, 187–192. doi:10.1145/3674399.3674445

[20] Bei Chen, Gaolei Li, Jun Wu, Jianhua Li, Mingzhe Chen, and Jiacheng Wang. 2026. Agentchain: Blockchain-empowered multi-agent coordination for trustworthy llm question-answering systems. *IEEE Transactions on Dependable and Secure Computing* (2026).

[21] Jiaju Chen, Yuxuan Lu, Xiaojie Wang, Huimin Zeng, Jing Huang, Jiri Gesi, Ying Xu, Bingsheng Yao, and Dakuo Wang. 2026. Multi-agent-as-judge: Aligning llm-agent-based automated evaluation with multi-dimensional human evaluation. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 17389–17416.

[22] Justin Chih-Yao Chen, Swarnadeep Saha, and Mohit Bansal. 2023. Reconcile: Round-table conference improves reasoning via consensus among diverse llms. *arXiv preprint arXiv:2309.13007* (2023).

[23] Kai Chen, Taihang Zhen, Hewei Wang, Kailai Liu, Xinfeng Li, Jing Huo, Tianpei Yang, Jinfeng Xu, Wei Dong, and Yang Gao. 2025. MedSentry: Understanding and Mitigating Safety Risks in Medical LLM Multi-Agent Systems. *arXiv preprint arXiv:2505.20824* (2025).

[24] Mengzhuo Chen, Junjie Wang, Fangwen Mu, Yawen Wang, Zhe Liu, Huanxiang Feng, and Qing Wang. 2026. Seeing the whole elephant: A benchmark for failure attribution in llm-based multi-agent systems. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 19888–19905.

[25] Tiejin Chen, Huaiyuan Yao, Jia Chen, Evangelos E Papalexakis, and Hua Wei. 2026. Every Response Counts: Quantifying Uncertainty of LLM-based Multi-Agent Systems through Tensor Decomposition. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 16204–16218.

[26] Weize Chen, Chenfei Yuan, Jiarui Yuan, Yusheng Su, Chen Qian, Cheng Yang, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2024. Beyond natural language: Llms leveraging alternative formats for enhanced reasoning and communication. *arXiv preprint arXiv:2402.18439* (2024).

[27] Ying-Jung Chen and Vijay Madisetti. 2025. Information Security, Ethics, and Integrity in LLM Agent Interaction. *Journal of Information Security* 16 (01 2025), 184–196. doi:10.4236/jis.2025.161010

[28] Hyeong Kyu Choi, Jerry Zhu, and Sharon Li. 2026. Debate or vote: Which yields better decisions in multi-agent large language models? *Advances in Neural Information Processing Systems* 38 (2026), 101732–101764.

[29] Seojin Choi and Geunseok Yang. 2025. Agentreport: A multi-agent llm approach for automated and reproducible bug report generation. *Applied Sciences* 15, 22 (2025), 11931.

[30] Jason A. Colquitt. 2001. On the Dimensionality of Organizational Justice: A Construct Validation of a Measure. *Journal of Applied Psychology* 86, 3 (2001), 386–400. doi:10.1037/0021-9010.86.3.386

[31] Mihaela Constantinescu and Muel Kaptein. 2025. Responsibility gaps, LLMs & organisations: Many agents, many levels, and many interactions.

[32] Mihaela Constantinescu and Muel Kaptein. 2025. Responsibility gaps, LLMs & organizations: Many agents, many levels, and many interactions.

[33] Yue Cui, Liuyi Yao, Zitao Li, Yaliang Li, Bolin Ding, and Xiaofang Zhou. 2025. Efficient Leave-one-out Approximation in LLM Multi-agent Debate Based on Introspection. *arXiv preprint arXiv:2505.22192* (2025).

[34] D. B. Davis, J. Featherston, M. Fukuda, and H. U. Asuncion. 2017. Data Provenance for Multi-Agent Models. In *2017 IEEE 13th International Conference on e-Science*. IEEE, 39–48.

[35] José Antonio Siqueira de Cerqueira, Mamia Agbese, Rebekah Rousi, Nannan Xi, Juho Hamari, and Pekka Abrahamsson. 2024. Can We Trust AI Agents? A Case Study of an LLM-Based Multi-Agent System for Ethical AI. *arXiv preprint arXiv:2411.08881* (2024).

[36] Martijn de Vos, Akash Dhasade, Jade Garcia Bourrée, Anne-Marie Kermarrec, Erwan Le Merrer, Benoit Rottembourg, and Gilles Tredan. 2024. Fairness auditing with multi-agent collaboration. *arXiv preprint arXiv:2402.08522* (2024).

[37] Edoardo Debenedetti, Jie Zhang, Mislav Balunovic, Luca Beurer-Kellner, Marc Fischer, and Florian Tramèr. 2024. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *Advances in Neural Information Processing Systems* 37 (2024), 82895–82920.

[38] Zehang Deng, Yongjian Guo, Changzhou Han, Wanlun Ma, Junwu Xiong, Sheng Wen, and Yang Xiang. 2025. Ai agents under threat: A survey of key security challenges and future pathways. *Comput. Surveys* 57, 7 (2025), 1–36.

[39] Liming Dong, Qinghua Lu, and Liming Zhu. 2024. Agentops: Enabling observability of llm agents. *arXiv preprint arXiv:2411.05285* (2024).

[40] Yilun Du, Shuang Li, Antonio Torralba, Joshua B Tenenbaum, and Igor Mordatch. 2023. Improving factuality and reasoning in language models through multiagent debate. *arXiv preprint arXiv:2305.14325* (2023).

[41] Zhihua Duan and Jialin Wang. 2025. Enhancing multi-agent consensus through third-party llm integration: Analyzing uncertainty and mitigating hallucinations in large language models. In *2025 8th International Conference on Advanced Algorithms and Control Engineering (ICAACE)*. IEEE, 2222–2227.

[42] Sana Ebrahimi, Mohsen Dehghankar, and Abolfazl Asudeh. 2025. An Adversary-Resistant Multi-Agent LLM System via Credibility Scoring. *arXiv preprint arXiv:2505.24239* (2025).

[43] Sana Ebrahimi, Nima Shahbazi, and Abolfazl Asudeh. 2024. REQUAL-LM: Reliability and Equity through Aggregation in Large Language Models. In *NAACL-HLT (Findings)*.

[44] Nicholas Edwards and Sebastian Schuster. 2026. Ask or assume? uncertainty-aware clarification-seeking in coding agents. *arXiv preprint arXiv:2603.26233* (2026).

[45] Yanai Elazar, Nora Kassner, Shauli Ravfogel, et al. 2021. Measuring and Improving Consistency in Pretrained Language Models. *Transactions of the Association for Computational Linguistics* 9 (2021), 1012–1031.

[46] Cornelius Emde, Anmol Goel, Sangdoo Yun, Seong Joon Oh, and Martin Gubri. 2026. Lost in Communication: Uncertainty Propagation in Multi-Agent Systems. In *ICML 2026 Statistical Frameworks for Uncertainty in Agentic Systems*.

[47] Wei Fan, JinYi Yoon, and Bo Ji. 2026. imad: Intelligent multi-agent debate for efficient and accurate llm inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 29403–29411.

[48] Yu Feng, Phu Mon Htut, Zheng Qi, Wei Xiao, Manuel Mager, Nikolaos Pappas, Kishaloy Halder, Yang Li, Yassine Benajiba, and Dan Roth. 2024. Rethinking LLM uncertainty: A multi-agent approach to estimating black-box model uncertainty. *arXiv preprint arXiv:2412.09572* (2024).

[49] Jose L Garcia, Karolina Hajkova, Maria Marchenko, and Carlos Miguel Patiño. 2025. Reproducibility Study of Cooperation, Competition, and Maliciousness: LLM-Stakeholders Interactive Negotiation. *arXiv preprint arXiv:2502.16242* (2025).

[50] Roxana Geambasu, Mariana Raykova, Pierre Tholoniat, Trishita Tiwari, Lillian Tsai, and Wen Zhang. 2026. Engineering robustness into personal agents with the AI workflow store. *arXiv preprint arXiv:2605.10907* (2026).

[51] Ashkan Golgoon, Khashayar Filom, and Arjun Ravi Kannan. 2024. Mechanistic interpretability of large language models with applications to the financial services industry. In *Proceedings of the 5th ACM International Conference on AI in Finance*. 660–668.

[52] Kai Greshake, Sahar Abdelnabi, Shailesh Mishra, Christoph Endres, Thorsten Holz, and Mario Fritz. 2023. Not what you've signed up for: Compromising real-world llm-integrated applications with indirect prompt injection. In *Proceedings of the 16th ACM workshop on artificial intelligence and security*. 79–90.

[53] Boyuan Guan, Jason Liu, Yanzhao Wu, and Kiavash Bahreini. 2026. Exploring Robust Multi-Agent Workflows for Environmental Data Management. *arXiv preprint arXiv:2604.01647* (2026).

[54] Yanxing Guo, Zihao Zheng, Fangzhou Wu, Ling Liang, Lin Bao, Zongwei Wang, and Yimao Cai. 2026. DynaGraph: Lightweight Multi-Model Interaction Framework via Dynamic Topological Reconfiguration. *arXiv preprint arXiv:2605.29511* (2026).

[55] Benjamin Haibe-Kains, George Alexandru Adam, Ahmed Hosny, Farnoosh Khodakarami, Massive Analysis Quality Control (MAQC) Society Board of Directors Shraddha Thakkar 35 Kusko Rebecca 36 Sansone Susanna-Assunta 37 Tong Weida 35 Wolfinger Russ D. 38 Mason Christopher E. 39 Jones Wendell 40 Dopazo Joaquin 41 Furlanello Cesare 42, Levi Waldron, Bo Wang, Chris McIntosh, Anna Goldenberg, Anshul Kundaje, et al. 2020. Transparency and reproducibility in artificial intelligence. *Nature* 586, 7829 (2020), E14–E16.

[56] Shawn He, Surangika Ranathunga, Stephen Cranefield, and Bastin Tony Roy Savarimuthu. 2024. Norm violation detection in multi-agent systems using large language models: A pilot study. In *International Workshop on Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems*. Springer, 146–159.

[57] Nathan Herr, Fernando Acero, Roberta Raileanu, María Pérez-Ortiz, and Zhibin Li. 2024. Are large language models strategic decision makers? a study of performance and bias in two-player non-zero-sum games. *arXiv preprint arXiv:2407.04467* (2024).

[58] Jinwei Hu, Yi Dong, Shuang Ao, Zhuoyun Li, Boxuan Wang, Lokesh Singh, Guangliang Cheng, Sarvapali D Ramchurn, and Xiaowei Huang. 2025. Position: Towards a responsible llm-empowered multi-agent systems. *arXiv preprint arXiv:2502.01714* (2025).

[59] Jinwei Hu, Yi Dong, Zhengtao Ding, and Xiaowei Huang. 2025. Enhancing Robustness of LLM-Driven Multi-Agent Systems through Randomized Smoothing. *arXiv preprint arXiv:2507.04105* (2025).

[60] Tianyu Hu, Zhen Tan, Song Wang, Huaizhi Qu, and Tianlong Chen. 2026. Multi-agent debate for llm judges with adaptive stability detection. *Advances in Neural Information Processing Systems* 38 (2026), 46504–46540.

[61] Yun Hua, Haosheng Chen, Shiqin Wang, Wenhao Li, Xiangfeng Wang, and Jun Luo. 2026. Shapley-Coop: Credit assignment for emergent cooperation in self-interested LLM agents. *Advances in Neural Information Processing Systems* 38 (2026), 88675–88702.

[62] Jiatan Huang, Mingchen Li, Ziming Li, Sunjae Kwon, Hong Yu, and Chuxu Zhang. 2026. Counterfactual Graph for Multi-Agent LLM Calibration. *arXiv preprint arXiv:2605.30653* (2026).

[63] Jen-tse Huang, Jiaxu Zhou, Tailin Jin, Xuhui Zhou, Zixi Chen, Wenxuan Wang, Youliang Yuan, Maarten Sap, and Michael R Lyu. 2024. On the resilience of multi-agent systems with malicious agents. *arXiv preprint arXiv:2408.00989* (2024).

[64] Muhua Huang, Qinlin Zhao, Xiaoyuan Yi, and Xing Xie. 2025. On the Dynamics of Multi-Agent LLM Communities Driven by Value Diversity. *arXiv preprint arXiv:2512.10665* (2025).

[65] Yeonjun In, Mehrab Tanjim, Jayakumar Subramanian, Sungchul Kim, Uttaran Bhattacharya, Wonjoong Kim, Sangwu Park, Somdeb Sarkhel, and Chanyoung Park. 2026. Rethinking failure attribution in multi-agent systems: A multi-perspective benchmark and evaluation. *arXiv preprint arXiv:2603.25001* (2026).

[66] Saeid Jamshidi. 2026. Collective Hallucination in Multi-Agent LLMs: Modeling and Defense. *arXiv preprint arXiv:2606.07941* (2026).

[67] Bo Jiang. 2026. DiscoUQ: Structured Disagreement Analysis for Uncertainty Quantification in LLM Agent Ensembles. *arXiv preprint arXiv:2603.20975* (2026).

[68] Jiechuan Jiang and Zongqing Lu. 2019. Learning fairness in multi-agent systems. *Advances in Neural Information Processing Systems* 32 (2019).

[69] Pengyi Jiang, Xiaoguang Zhu, and Quanyan Zhu. 2026. Semantic Cooperative Games for Contribution Attribution in LLM-Based Multi-Agent Systems. *arXiv preprint arXiv:2607.18255* (2026).

[70] Alibek T Kaliyev and Artem Maryanskyy. 2026. How Much Coordination Gain Is Real? A Paired Noise-Floor Protocol for Multi-Agent LLM Benchmarks. *arXiv preprint arXiv:2606.20695* (2026).

[71] Vira Kasprova, Amruta Parulekar, Abdulrahman AlRabah, Krishna Agaram, Ritwik Garg, Sagar Jha, Nimet Beyza Bozdag, and Dilek Hakkani-Tur. 2026. Too Polite to Disagree: Understanding Sycophancy Propagation in Multi-Agent Systems. *arXiv preprint arXiv:2604.02668* (2026).

[72] Dezhang Kong, Shi Lin, Zhenhua Xu, Zhebo Wang, Minghao Li, Yufeng Li, Yilun Zhang, Zeyang Sha, Yuyuan Li, Changting Lin, et al. 2025. A Survey of LLM-Driven AI Agent Communication: Protocols, Security Risks, and Defense Countermeasures. *arXiv preprint arXiv:2506.19676* (2025).

[73] Maya Kruse, Majid Afshar, Saksham Khatwani, Anoop Mayampurath, Guanhua Chen, and Yanjun Gao. 2025. Simple yet effective: An information-theoretic approach to multi-LLM uncertainty quantification. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 30481–30492.

[74] Xian Yeow Lee, Shunichi Akatsuka, Lasitha Vidyaratne, Aman Kumar, Ahmed Farahat, and Chetan Gupta. 2025. Reliable Decision-Making for Multi-Agent LLM Systems. *arXiv preprint arXiv:2406.04092* (2025).

[75] Yu Lei, Hao Liu, Chengxing Xie, Songjia Liu, Zhiyu Yin, Canyu Chen, Guohao Li, Philip Torr, and Zhen Wu. 2024. Fairmindsim: Alignment of behavior, emotion, and belief in humans and llm agents amid ethical dilemmas. *arXiv preprint arXiv:2410.10398* (2024).

[76] Dawei Li, Zhen Tan, Peijia Qian, Yifan Li, Kumar Chaudhary, Lijie Hu, and Jiayi Shen. 2025. SMoA: Improving Multi-agent Large Language Models with S parse M ixture-o f-A gents. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 54–65.

[77] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *Advances in Neural Information Processing Systems* 36 (2023), 51991–52008.

[78] Huao Li, Yu Quan Chong, Simon Stepputtis, Joseph Campbell, Dana Hughes, Michael Lewis, and Katia Sycara. 2023. Theory of mind for multi-agent collaboration via large language models. *arXiv preprint arXiv:2310.10701* (2023).

[79] Jiayi Li, Xiao Liu, and Yansong Feng. 2026. From single to societal: Analyzing persona-induced bias in multi-agent interactions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 31609–31617.

[80] Jingyue Li and André Storhaug. 2026. Reproducible, Explainable, and Effective Evaluations of Agentic AI for Software Engineering. In *Proceedings of the 34th ACM International Conference on the Foundations of Software Engineering*. 1501–1506.

[81] Junyou Li, Qin Zhang, Yangbin Yu, Qiang Fu, and Deheng Ye. 2024. More agents is all you need. *arXiv preprint arXiv:2402.05120* (2024).

[82] Tianlin Li, Qian Liu, Tianyu Pang, Chao Du, Qing Guo, Yang Liu, and Min Lin. 2024. Purifying large language models by ensembling a small language model. *arXiv preprint arXiv:2402.14845* (2024).

[83] Tianxiao Li, Yixing Ma, Haiquan Wen, Zhenglin Huang, Qianyu Zhou, Zeyu Fu, and Guangliang Cheng. 2026. Safe Multi-Agent Behavior Must Be Maintained, Not Merely Asserted: Constraint Drift in LLM-Based Multi-Agent Systems. *arXiv preprint arXiv:2605.10481* (2026).

[84] Yanming Li, Xuelin Zhang, WenJie Lu, Ziye Tang, Maodong Wu, Haotian Luo, Tongtong Wu, Zijie Peng, Hongze Mi, Yibo Feng, et al. 2026. Who deserves the reward? sharp: Shapley credit-based optimization for multi-agent system. *arXiv preprint arXiv:2602.08335* (2026).

[85] Tian Liang, Zhiwei He, Wenxiang Jiao, Xing Wang, Yan Wang, Rui Wang, Yujiu Yang, Shuming Shi, and Zhaopeng Tu. 2023. Encouraging divergent thinking in large language models through multi-agent debate. *arXiv preprint arXiv:2305.19118* (2023).

[86] Xuechen Liang, Yangfan He, Meiling Tao, Yinghui Xia, Jianhui Wang, Tianyu Shi, Jun Wang, and JingSong Yang. 2024. Cmat: A multi-agent collaboration tuning framework for enhancing small language models. *arXiv preprint arXiv:2404.01663* (2024).

[87] Hongzhan Lin, Yang Deng, Yuxuan Gu, Wenxuan Zhang, Jing Ma, See-Kiong Ng, and Tat-Seng Chua. 2025. Fact-audit: An adaptive multi-agent framework for dynamic fact-checking evaluation of large language models. *arXiv preprint arXiv:2502.17924* (2025).

[88] Zachary C Lipton. 2018. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue* 16, 3 (2018), 31–57.

[89] Chengzhi Liu, Yichen Guo, Yepeng Liu, Yuzhe Yang, Qianqi Yan, Xuandong Zhao, Wenyue Hua, Sheng Liu, Sharon Li, Yuheng Bu, et al. 2026. Auditing agent harness safety. *arXiv preprint arXiv:2605.14271* (2026).

[90] Zijun Liu, Yanzhe Zhang, Peng Li, Yang Liu, and Diyi Yang. 2023. Dynamic llm-agent network: An llm-agent collaboration framework with agent team optimization. *arXiv preprint arXiv:2310.02170* (2023).

[91] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. 2023. Self-refine: Iterative refinement with self-feedback. *Advances in Neural Information Processing Systems* 36 (2023), 46534–46594.

[92] Maeve Madigan, Parameswaran Kamalaruban, Glenn Moynihan, Tom Kempton, David Sutton, and Stuart Burrell. 2025. Emergent Bias and Fairness in Multi-Agent Decision Systems. *arXiv preprint arXiv:2512.16433* (2025).

[93] Andreas Madsen, Siva Reddy, and Sarath Chandar. 2022. Post-hoc interpretability for neural nlp: A survey. *Comput. Surveys* 55, 8 (2022), 1–42.

[94] Wenji Mao and Jonathan Gratch. 2012. Modeling social causality and responsibility judgment in multi-agent interactions. *Journal of Artificial Intelligence Research* 44 (2012), 223–273.

[95] Kawin Mayilvaghanan, Siddhant Gupta, and Ayush Kumar. 2026. Counterfactual Fairness Evaluation of LLM-Based Contact Center Agent Quality Assurance System. In *Findings of the Association for Computational Linguistics: ACL 2026*. 37914–37952.

[96] Parsa Mazaheri. 2026. REPOT: Recoverable Program-of-Thought via Checkpoint Repair. *arXiv preprint arXiv:2605.30052* (2026).

[97] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)* 54, 6 (2021), 1–35.

[98] Bahar Memarian and Tenzin Doleck. 2023. Fairness, Accountability, Transparency, and Ethics (FATE) in Artificial Intelligence (AI) and higher education: A systematic review. *Computers and Education: Artificial Intelligence* 5 (2023), 100152. doi:10.1016/j.caeai.2023.100152

[99] Imran Mirza, Cole Huang, Ishwara Vasista, Rohan Patil, Asli Akalin, Sean O'Brien, and Kevin Zhu. 2025. Malibu benchmark: Multi-agent llm implicit bias uncovered. *arXiv preprint arXiv:2507.01019* (2025).

[100] Jakob Mökander, Jonas Schuett, Hannah Rose Kirk, and Luciano Floridi. 2024. Auditing large language models: a three-layered approach. *AI and Ethics* 4, 4 (2024), 1085–1115.

[101] Claude Moran, Xiaokai Liao, and Henrik Norris. 2026. Trustworthy Multi-Agent LLM Collaboration: A Prototype Consistency and Explainability Framework for Secure Distributed Intelligence Systems. *Journal of Data Intelligence and AI Systems* 1, 1 (2026).

[102] Fuseini Mumuni and Alhassan Mumuni. 2025. Explainable artificial intelligence (XAI): from inherent explainability to large language models. *arXiv preprint arXiv:2501.09967* (2025).

[103] Rahul Nanda, Chandra Maddila, Smriti Jha, Euna Mehnaz Khan, Matteo Paltenghi, and Satish Chandra. 2026. Wink: Recovering from misbehaviors in coding agents. In *Proceedings of the 3rd ACM International Conference on AI-Powered Software*. 208–217.

[104] Thi-Nhung Nguyen, Linhao Luo, Amardeep Kaur, Rollin Omari, Tamas Abraham, Junae Kim, Thuy-Trang Vu, and Dinh Phung. 2025. The social cost of intelligence: Emergence, propagation, and amplification of stereotypical bias in multi-agent systems. *arXiv preprint arXiv:2510.10943* (2025).

[105] Yi Nian, Aojie Yuan, Haiyue Zhang, Jiate Li, and Yue Zhao. 2026. Auditable agents. *arXiv preprint arXiv:2604.05485* (2026).

[106] Maya Okawa. 2026. Emergence of Biased Consensus in Multi-Agent LLM Debates. *arXiv preprint arXiv:2608.02827* (2026).

[107] Pedro Oliveira, João da Cruz Pereira, and Paulo Novais. 2026. *Architectures for Agentic AI: Integrating Multi-Agent Systems, Reinforcement Learning, and LLMs for Autonomous Decision-Making*. Springer Nature.

[108] Barak Or. 2025. MTTR-A: Measuring Cognitive Recovery Latency in Multi-Agent Systems. *arXiv preprint arXiv:2511.20663* (2025).

[109] Junjun Pan, Yixin Liu, Rui Miao, Kaize Ding, Yu Zheng, Quoc Viet Hung Nguyen, Alan Wee-Chung Liew, and Shirui Pan. 2026. Explainable and fine-grained safeguarding of llm multi-agent systems via bi-level graph anomaly detection. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 30491–30506.

[110] Erik Pautsch, Tanmay Singla, Parv Kumar, Wenxin Jiang, Huiyun Peng, Behnaz Hassanshahi, Konstantin Läufer, George K Thiruvathukal, and James C Davis. 2025. Agenthub: A registry for discoverable, verifiable, and reproducible ai agents. *arXiv preprint arXiv:2510.03495* (2025).

[111] Pouya Pezeshkpour, Eser Kandogan, Nikita Bhutani, Sajjadur Rahman, Tom Mitchell, and Estevam Hruschka. 2024. Reasoning capacity in multi-agent systems: Limitations, challenges and human-centered solutions. *arXiv preprint arXiv:2402.01108* (2024).

[112] Chau Pham, Boyi Liu, Yingxiang Yang, Zhengyu Chen, Tianyi Liu, Jianbo Yuan, Bryan A Plummer, Zhaoran Wang, and Hongxia Yang. 2023. Let models speak ciphers: Multiagent debate through embeddings. *arXiv preprint arXiv:2310.06272* (2023).

[113] Giorgio Piatti, Zhijing Jin, Max Kleiman-Weiner, Bernhard Schölkopf, Mrinmaya Sachan, and Rada Mihalcea. 2024. Cooperate or collapse: Emergence of sustainable cooperation in a society of llm agents. *Advances in Neural Information Processing Systems* 37 (2024), 111715–111759.

[114] David MW Powers. 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061* (2020).

[115] Shihao Qi, Jie Ma, Rui Xing, Wei Guo, Xiao Huang, Zhitao Gao, Jianhao Deng, Jun Liu, Lingling Zhang, Bifan Wei, et al. 2026. Beyond Individual Intelligence: Surveying Collaboration, Failure Attribution, and Self-Evolution in LLM-based Multi-Agent Systems. *arXiv preprint arXiv:2605.14892* (2026).

[116] Chen Qian, Wei Liu, Hongzhang Liu, Nuo Chen, Yufan Dang, Jiahao Li, Cheng Yang, Weize Chen, Yusheng Su, Xin Cong, et al. 2023. Chatdev: Communicative agents for software development. *arXiv preprint arXiv:2307.07924* (2023).

[117] Chen Qian, Zihao Xie, Yifei Wang, Wei Liu, Yufan Dang, Zhuoyun Du, Weize Chen, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2024. Scaling Large-Language-Model-based Multi-Agent Collaboration. *arXiv preprint arXiv:2406.07155* (2024).

[118] Edward Raff, Michel Benaroch, Sagar Samtani, and Andrew L Farris. 2025. What Do Machine Learning Researchers Mean by "Reproducible"?. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 28671–28683.

[119] Inioluwa Raji, Peggy Xu, Colleen Honigsberg, and Daniel Ho. 2022. Outsider Oversight: Designing a Third Party Audit Ecosystem for AI Governance. 557–571. doi:10.1145/3514094.3534181

[120] Vignav Ramesh and Kenneth Li. 2025. Communicating Activations Between Language Model Agents. *arXiv preprint arXiv:2501.14082* (2025).

[121] Rajesh Ranjan, Shailja Gupta, and Surya Narayan Singh. 2025. Fairness in Agentic AI: A Unified Framework for Ethical and Equitable Multi-Agent System. *arXiv preprint arXiv:2502.07254* (2025).

[122] Shaina Raza, Ranjan Sapkota, Manoj Karkee, and Christos Emmanouilidis. 2025. Trism for agentic ai: A review of trust, risk, and security management in llm-based agentic multi-agent systems. *arXiv preprint arXiv:2506.04133* (2025).

[123] Adrián Romero-Garcés, Alejandro Hidalgo-Paniagua, Martín González-García, and Antonio Bandera. 2022. On Managing Knowledge for MAPE-K Loops in Self-Adaptive Robotics Using a Graph-Based Runtime Model. *Applied Sciences* 12, 17 (2022). doi:10.3390/app12178583

[124] Josh Rosen and Seth Rosen. 2026. From Agent Loops to Deterministic Graphs: Execution Lineage for Reproducible AI-Native Work. *arXiv preprint arXiv:2605.06365* (2026).

[125] Manish Sanwal. 2025. Layered chain-of-thought prompting for multi-agent llm systems: A comprehensive approach to explainable large language models. *arXiv preprint arXiv:2501.18645* (2025).

[126] Rohan Sequeira, Stavros Damianakis, Umar Iqbal, and Konstantinos Psounis. 2026. Agent-sentry: Bounding llm agents via execution provenance. *arXiv preprint arXiv:2603.22868* (2026).

[127] Jaineet Shah. 2026. Causal Agent Replay: Counterfactual Attribution for LLM-Agent Failures. *arXiv preprint arXiv:2606.08275* (2026).

[128] Lloyd S Shapley. 1951. Notes on the n-person game—ii: The value of an n-person game. *RAND Corporation, RM-670* (1951).

[129] Xu Shen, Qi Zhang, Song Wang, Zhen Tan, Xinyu Zhao, Laura Yao, Vaishnav Tadiparthi, Hossein Nourkhiz Mahjoub, Ehsan Moradi Pari, Kwonjoon Lee, et al. 2026. Metacognitive self-correction for multi-agent system via prototype-guided next-execution reconstruction. In *Findings of the*

Association for Computational Linguistics: ACL 2026. 23320–23337.

[130] Yoav Shoham and Kevin Leyton-Brown. 2008. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, USA.

[131] Tassio Ferenzini Martins Sirqueira, Marcelo Ladeira Viana, Francisco José Parreiras da Cunha, Ingrid Nunes, and Carlos José Pereira de Lucena. 2018. Data Provenance in Multi-Agent Systems: Relevance, Benefits and Research Opportunities. *International Journal of Metadata, Semantics and Ontologies* 13, 1 (2018), 9–19.

[132] Mohsen Sorkhpour, Abbas Yazdinejad, and Ali Dehghantanha. 2025. RedHit: Adaptive Red-Teaming of Large Language Models via Search, Reasoning, and Preference Optimization. In *Proceedings of the The First Workshop on LLM Security (LLMSEC)*, Leon Derczynski, Jekaterina Novikova, and Muhao Chen (Eds.). Association for Computational Linguistics, Vienna, Austria, 7–16. https://aclanthology.org/2025.llmsec-1.2/

[133] Renan Souza, Amal Gueroudji, Stephen DeWitt, Daniel Rosendo, Tirthankar Ghosal, Robert Ross, Prasanna Balaprakash, and Rafael Ferreira da Silva. 2025. PROV-AGENT: Unified Provenance for Tracking AI Agent Interactions in Agentic Workflows. In *2025 IEEE International Conference on eScience*. IEEE, 467–473.

[134] Junhao Su, Yuanliang Wan, Junwei Yang, Hengyu Shi, Tianyang Han, Yurui Qiu, and Junfeng Luo. 2026. Failure makes the agent stronger: Enhancing accuracy through structured reflection for reliable tool interactions. In *Findings of the Association for Computational Linguistics: ACL 2026*. 12712–12734.

[135] Kangkang Sun, Jun Wu, Jianhua Li, Minyi Guo, Xiuzhen Che, and Jianwei Huang. 2026. CoE: Collaborative Entropy for Uncertainty Quantification in Agentic Multi-LLM Systems. *arXiv preprint arXiv:2603.28360* (2026).

[136] Lijun Sun, Yijun Yang, Qiqi Duan, Yuhui Shi, Chao Lyu, Yu-Cheng Chang, Chin-Teng Lin, and Yang Shen. 2025. Multi-agent coordination across diverse applications: A survey. *arXiv preprint arXiv:2502.14743* (2025).

[137] Xiaolin Sun, Zixuan Liu, Yibin Hu, and Zizhan Zheng. 2026. Insider Attacks in Multi-Agent LLM Consensus Systems. *arXiv preprint arXiv:2605.08268* (2026).

[138] Sagar Tamang and Dibya Jyoti Bora. 2025. Enforcement Agents: Enhancing Accountability and Resilience in Multi-Agent AI Frameworks. *arXiv preprint arXiv:2504.04070* (2025).

[139] Qian Tan, Lei Jiang, Yuting Zeng, Shuoyang Ding, and Xiaohua Xu. 2026. Mitigating Cultural Bias in LLMs via Multi-Agent Cultural Debate. In *Findings of the Association for Computational Linguistics: ACL 2026*. 8600–8612.

[140] Yichen Tang, Weihang Su, Yujia Zhou, Yiqun Liu, Min Zhang, Shaoping Ma, and Qingyao Ai. 2025. Augmenting Multi-Agent Communication with State Delta Trajectory. *arXiv preprint arXiv:2506.19209* (2025).

[141] Yu Tian, Xiao Yang, Jingyuan Zhang, Yinpeng Dong, and Hang Su. 2023. Evil geniuses: Delving into the safety of llm-based agents. *arXiv preprint arXiv:2311.11855* (2023).

[142] Arne Tillmann. 2025. Literature Review Of Multi-Agent Debate For Problem-Solving. *arXiv preprint arXiv:2506.00066* (2025).

[143] Khanh-Tung Tran, Dung Dao, Minh-Duong Nguyen, Quoc-Viet Pham, Barry O'Sullivan, and Hoang D Nguyen. 2025. Multi-Agent Collaboration Mechanisms: A Survey of LLMs. *arXiv preprint arXiv:2501.06322* (2025).

[144] Bhanu Prakash Vangala, Ali Adibifar, Ashish Gehani, and Tanu Malik. 2025. AI-generated code is not reproducible (yet): an empirical study of dependency gaps in LLM-based coding agents. *arXiv preprint arXiv:2512.22387* (2025).

[145] Guangya Wan, Mingyang Ling, Xiaoqi Ren, Rujun Han, Sheng Li, and Zizhao Zhang. 2026. Compass: Enhancing agent long-horizon reasoning with evolving context. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 3360–3380.

[146] Junlin Wang, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. 2024. Mixture-of-Agents Enhances Large Language Model Capabilities. *arXiv preprint arXiv:2406.04692* (2024).

[147] Peiran Wang, Ying Li, and Yuan Tian. 2026. Aligning Provenance with Authorization: A Dual-Graph Defense for LLM Agents. *arXiv preprint arXiv:2605.26497* (2026).

[148] Qianli Wang, Tatiana Anikina, Nils Feldhus, Josef Genabith, Leonhard Hennig, and Sebastian Möller. 2024. LLMCheckup: Conversational examination of large language models via interpretability tools and self-explanations. In *Proceedings of the Third Workshop on Bridging Human–Computer Interaction and Natural Language Processing*. 89–104.

[149] Qian Wang, Tianyu Wang, Zhenheng Tang, Qinbin Li, Nuo Chen, Jingsheng Liang, and Bingsheng He. 2025. MegaAgent: A large-scale autonomous LLM-based multi-agent system without predefined SOPs. In *Findings of the Association for Computational Linguistics: ACL 2025*. 4998–5036.

[150] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171* (2022).

[151] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2022. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171* (2022).

[152] Xinru Wang, Ming Yin, Eunyee Koh, and Mustafa Doga Dogan. 2026. XAgen: An Explainability Tool for Identifying and Correcting Failures in Multi-Agent Workflows. In *Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems*. 1–11.

[153] Yuntao Wang, Yanghe Pan, Zhou Su, Yi Deng, Quan Zhao, Linkang Du, Tom H Luan, Jiawen Kang, and Dusit Niyato. 2025. Large model based agents: State-of-the-art, cooperation paradigms, security and privacy, and future trends. *IEEE Communications Surveys & Tutorials* (2025).

[154] Yiqi Wang, Jiaqi Zhang, Taotao Cai, Zirui Liu, Qingqiang Sun, Zequn Sun, Zhangkai Wu, Manqing Dong, Mingkai Zhang, Xuefei Yin, et al. 2026. From Agent Traces to Trust: A Survey of Evidence Tracing and Execution Provenance in LLM Agents. *arXiv e-prints* (2026), arXiv–2606.

[155] Hilde Weerts, Miroslav Dudík, Richard Edgar, Adrin Jalali, Roman Lutz, and Michael Madaio. 2023. Fairlearn: Assessing and improving fairness of AI systems. *Journal of Machine Learning Research* 24, 257 (2023), 1–8.

[156] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[157] Maranke Wieringa. 2020. What to Account for When Accounting for Algorithms: A Systematic Literature Review on Algorithmic Accountability. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. 1–18. doi:10.1145/3351095.3372833

[158] Lorenz Wiesmeier, Matthias Busch, Marius Tacke, Kevin Linka, Christian Cyron, and Roland Aydin. 2026. Adversarial robustness of LLM-based multi-agent systems for engineering problems. *Frontiers in Artificial Intelligence* 9 (2026), 1784484.

[159] Rebecca Williams, Richard Cloete, Jennifer Cobbe, Caitlin Cottrill, Peter Edwards, Milan Markovic, Iman Naja, Frances Ryan, Jatinder Singh, and Wei Pang. 2022. From Transparency to Accountability of Intelligent Systems: Moving Beyond Aspirations. *Data & Policy* 4 (2022), e7. doi:10.1017/dap.2021.37

[160] Michael Wooldridge. 2009. *An introduction to multiagent systems*. John wiley & sons.

[161] Qingyun Wu, Gagan Bansal, Jieyu Zhang, Yiran Wu, Shaokun Zhang, Erkang Zhu, Beibin Li, Li Jiang, Xiaoyun Zhang, and Chi Wang. 2023. Autogen: Enabling next-gen llm applications via multi-agent conversation framework. *arXiv preprint arXiv:2308.08155* (2023).

[162] Zengqing Wu and Takayuki Ito. 2025. The hidden strength of disagreement: Unraveling the consensus-diversity tradeoff in adaptive multi-agent systems. *arXiv preprint arXiv:2502.16565* (2025).

[163] Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences* 68, 2 (2025), 121101.

[164] Boming Xia, Liming Zhu, Erdun Gao, Qinghua Lu, Minhui Xue, and Dino Sejdinovic. 2026. Uncertainty propagation in llm-based systems. *arXiv preprint arXiv:2604.23505* (2026).

[165] Yihan Xia, Taotao Wang, Shengli Zhang, Zhangyuhua Weng, Bin Cao, and Soung Chang Liew. 2026. Hivemind: Contribution-guided online prompt optimization of llm multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 29767–29774.

[166] Yizhe Xie, Congcong Zhu, Xinyue Zhang, Minghao Wang, Chi Liu, Minglu Zhu, and Tianqing Zhu. 2025. Who's the Mole? Modeling and Detecting Intention-Hiding Malicious Agents in LLM-Based Multi-Agent Systems. *arXiv preprint arXiv:2507.04724* (2025).

[167] Zhentao Xie, Jiabao Zhao, Yilei Wang, Jinxin Shi, Yanhong Bai, Xingjiao Wu, and Liang He. 2024. Mindscope: Exploring cognitive biases in large language models through multi-agent systems. *arXiv preprint arXiv:2410.04452* (2024).

[168] Zhenjie Xu, Wenqing Chen, Yi Tang, Xuanying Li, Cheng Hu, Zhixuan Chu, Kui Ren, Zibin Zheng, and Zhichao Lu. 2025. Mitigating social bias in large language models: A multi-objective approach within a multi-agent framework. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 25579–25587.

[169] Bingyu Yan, Xiaoming Zhang, Litian Zhang, Lian Zhang, Ziyi Zhou, Dezhuang Miao, and Chaozhuo Li. 2025. Beyond self-talk: A communication-centric survey of llm-based multi-agent systems. *arXiv preprint arXiv:2502.14321* (2025).

[170] John Yan, Michael Yu, Yuqi Sun, Alexander Duffy, Tyler Marques, and Matthew Lyle Olson. 2026. Data-Centric Interpretability for LLM-based Multi-Agent Reinforcement Learning. *arXiv preprint arXiv:2602.05183* (2026).

[171] Yingxuan Yang, Huacan Chai, Shuai Shao, Yuanyi Song, Siyuan Qi, Renting Rui, and Weinan Zhang. 2025. Agentnet: Decentralized evolutionary coordination for llm-based multi-agent systems. *arXiv preprint arXiv:2504.00587* (2025).

[172] Yi Yang, Yitong Ma, Hao Feng, Yiming Cheng, and Zhu Han. 2025. Minimizing Hallucinations and Communication Costs: Adversarial Debate and Voting Mechanisms in LLM-Based Multi-Agents. *Applied Sciences* 15 (03 2025), 3676. doi:10.3390/app15073676

[173] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

[174] Rui Ye, Xiangrui Liu, Qimin Wu, Xianghe Pang, Zhenfei Yin, Lei Bai, and Siheng Chen. 2025. X-MAS: Towards Building Multi-Agent Systems with Heterogeneous LLMs. *arXiv preprint arXiv:2505.16997* (2025).

[175] Miao Yu, Fanci Meng, Xinyun Zhou, Shilong Wang, Junyuan Mao, Linsey Pan, Tianlong Chen, Kun Wang, Xinfeng Li, Yongfeng Zhang, et al. 2025. A survey on trustworthy llm agents: Threats and countermeasures. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 6216–6226.

[176] Murong Yue. 2025. A survey of large language model agents for question answering. *arXiv preprint arXiv:2503.19213* (2025).

[177] Thomas Zeng, Shuibai Zhang, Shutong Wu, Christian Classen, Daewon Chae, Ethan Ewer, Minjae Lee, Heeju Kim, Wonjun Kang, Jackson Kunde, et al. 2025. Versaprm: Multi-domain process reward model via synthetic reasoning data. *arXiv preprint arXiv:2502.06737* (2025).

[178] Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. Injecagent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics: ACL 2024*. 10471–10506.

[179] Boxuan Zhang, Jianing Zhu, Zeru Shi, Dongfang Liu, and Ruixiang Tang. 2026. Agentforesight: Online auditing for early failure prediction in multi-agent systems. *arXiv preprint arXiv:2605.08715* (2026).

[180] Chongjie Zhang and Julie A Shah. 2014. Fairness in multi-agent sequential decision-making. *Advances in Neural Information Processing Systems* 27 (2014).

[181] Guibin Zhang, Junhao Wang, Junjie Chen, Wangchunshu Zhou, Kun Wang, and Shuicheng Yan. 2025. AgenTracer: Who Is Inducing Failure in the LLM Agentic Systems? *arXiv preprint arXiv:2509.03312* (2025).

[182] Hangfan Zhang, Zhiyao Cui, Jianhao Chen, Xinrun Wang, Qiaosheng Zhang, Zhen Wang, Dinghao Wu, and Shuyue Hu. 2025. Stop Overvaluing Multi-Agent Debate–We Must Rethink Evaluation and Embrace Model Heterogeneity. *arXiv preprint arXiv:2502.08788* (2025).

[183] Heng Zhang, Yuling Shi, Xiaodong Gu, Haochen You, Zijian Zhang, Lubin Gan, Yilei Yuan, and Jin Huang. 2025. GraphTracer: Graph-Guided Failure Tracing in LLM Agents for Robust Multi-Turn Deep Search. *arXiv e-prints* (2025), arXiv–2510.

[184] Jie M Zhang, Mark Harman, Lei Ma, and Yang Liu. 2020. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* 48, 1 (2020), 1–36.

[185] Miao Zhang, Junsik Kim, Siyuan Xiang, Jian Gao, and Cheng Cao. 2026. Dynamic Role Assignment for Multi-Agent Debate. *arXiv preprint arXiv:2601.17152* (2026).

[186] Man Zhang, Tao Yue, and Yihua He. 2026. Managing Uncertainty in LLM-based Multi-Agent System Operation. *arXiv preprint arXiv:2602.23005* (2026).

[187] Qian Zhang, Jinyi Liu, Yan Zheng, Hebin Liang, and Lanjun Wang. 2026. Key decision-makers in multi-agent debates: who holds the power?. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 40. 29883–29891.

[188] Shaokun Zhang, Ming Yin, Jieyu Zhang, Jiale Liu, Zhiguang Han, Jingyang Zhang, Beibin Li, Chi Wang, Huazheng Wang, Yiran Chen, et al. 2025. Which agent causes task failures and when? on automated failure attribution of llm multi-agent systems. *arXiv preprint arXiv:2505.00212* (2025).

[189] Yusen Zhang, Ruoxi Sun, Yanfei Chen, Tomas Pfister, Rui Zhang, and Sercan Ö Arik. 2024. Chain of Agents: Large Language Models Collaborating on Long-Context Tasks. *arXiv preprint arXiv:2406.02818* (2024).

[190] Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for Large Language Models: A Survey. *ACM Trans. Intell. Syst. Technol.* 15, 2, Article 20 (Feb. 2024), 38 pages. doi:10.1145/3639372

[191] Yuxuan Zhao, Sijia Chen, and Ningxin Su. 2026. When Does Multi-Agent Collaboration Help? An Entropy Perspective. *arXiv preprint arXiv:2602.04234* (2026).

[192] Yiyang Zhao, Zhuo Zhang, Qingxuan Le, Lizhen Qu, and Zenglin Xu. 2026. Beyond Goodhart's Law: A Dynamic Benchmark for Evaluating Compliance in Multi-Agent Systems. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*. 10338–10347.

[193] Heng Zhou, Hejia Geng, Xiangyuan Xue, Li Kang, Yiran Qin, Zhiyong Wang, Zhenfei Yin, and Lei Bai. 2025. Reso: A reward-driven self-organizing llm-based multi-agent system for reasoning tasks. *arXiv preprint arXiv:2503.02390* (2025).

[194] Pengcheng Zhou, Yinglun Feng, Halimulati Julaiti, and Zhongliang Yang. 2025. Why do AI agents communicate in human language? *arXiv preprint arXiv:2506.02739* (2025).

[195] Yihao Zhou, Tianjian Xia, Lu Ren, Jiajing Liao, Feng Chang, and Xinxin Huang. 2026. SwarmProbe: Explainability Analysis of Multi-Agent Systems via Emergent Behavior Probing. (2026).

[196] Chenyang Zhu, Spencer Hong, Jingyu Wu, Kushal Chawla, Yuhui Tang, Youbing Yin, Nathan Wolfe, Erin Babinsky, and Daben Liu. 2026. Raffles: Reasoning-based attribution of faults for llm systems. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*. 7659–7688.

[197] Judy Zhu, Dhari Gandh, Himanshu Joshi, Ahmad Rezaie Mianroodi, Sedef Akinli Kocak, and Dhanesh Ramachandran. 2026. Interpreting Agentic Systems: Beyond Model Explanations to System-Level Accountability. *arXiv preprint arXiv:2601.17168* (2026).

[198] Taiyu Zhu, Yifan Wu, Weilin Jin, Ying Li, and Gang Huang. 2026. StepFinder: A Temporal Semantic Framework for Failure Attribution in Multi-Agent Systems. *arXiv preprint arXiv:2606.03467* (2026).

[199] Xiaochen Zhu, Caiqi Zhang, Yizhou Chi, Tom Stafford, Nigel Collier, and Andreas Vlachos. 2026. Demystifying multi-agent debate: The role of confidence and diversity. In *Findings of the Association for Computational Linguistics: ACL 2026*. 33909–33930.

[200] Mingchen Zhuge, Wenyi Wang, Louis Kirsch, Francesco Faccio, Dmitrii Khizbullin, and Jürgen Schmidhuber. 2024. Gptswarm: Language agents as optimizable graphs. In *Forty-first International Conference on Machine Learning*.

Appendix

A Selection Principles of the Papers

We survey research on *LLM-powered multi-agent systems (LLM–MAS)* published or publicly archived since 2023. Our focus is on works that design, evaluate, or analyze multi-agent systems where large language models are primary reasoning or coordination components, and that contribute to at least one dimension of *responsibility* (reliability, accountability, transparency, fairness and ethics). We queried major digital libraries (ACL Anthology, arXiv, ACM DL, IEEE Xplore) using combinations of: *multi-agent, agentic, LLM agents, cooperative agents, debate/consensus, planner–executor, adjudicator, verifier, responsibility, reliability, explainability, transparency, accountability, fairness, auditability*. We complemented this with citation chaining (backward/forward) from seed papers and recent surveys to capture adjacent or very recent work. We include a paper when it proposes or empirically studies a multi-agent configuration with at least two interacting or an ensemble of LLM-based agents, reports methods, analyses, or evaluations that address at least one responsibility dimension, and provides sufficient methodological detail—covering models, prompts/protocols, datasets and tasks, or open-source artifacts—to allow meaningful assessment and, where feasible, reproduction. Within the eligible set, we highlight especially recent contributions (mid-2024–2026) that introduce new coordination protocols, evaluation suites for responsibility, or mechanisms for verification/auditing, and we synthesize their implications across our taxonomy. We do not provide a comprehensive review of MAS *security-only* literature, robotics-only multi-robot systems without LLM reasoning, or classical multi-agent planning without LLMs. Those areas are well covered elsewhere [27, 38, 72, 122, 153].

B Related Work

As interest in LLM-powered multi-agent systems grows, core challenges around scaling, communication, and collective decision-making have drawn sustained attention. Several recent surveys illuminate important pieces of this space. For fundamentals of LLM agents, their general architecture and key components we refer readers to Wang et al. [153], Xi et al. [163]. Tillmann [142] examine how scalability, communication structure, and decision processes shape performance; by synthesizing work on agent profiles, message topologies, and decision pipelines, they offer a careful analysis of scaling behavior and its pitfalls. Complementing this, Kong et al. [72] formalize agent communication, outlines its life cycle, and analyzes vulnerabilities and security risks alongside defense mechanisms. Focusing on a specific application family, Yue [176] review LLM agents for question answering across planning, query understanding,

Table 2. Comparison with Prior Surveys Scope

Surveys	LLM-MAS	Reliability	Transparency	Accountability	Fairness and Ethics
Memarian and Doleck [98]	×	×	✓	✓	✓
Zhao et al. [190]	×	×	✓	×	×
Raza et al. [122]	✓	✓	✓	✓	×
Constantinescu and Kaptein [32]	✓	×	✓	✓	×
Kong et al. [72]	✓	✓	×	✓	×
Yu et al. [175]	✓	✓	×	×	✓
Deng et al. [38]	✓	×	✓	×	×
This Survey	✓	✓	✓	✓	✓

retrieval, and answer synthesis, highlighting techniques for managing reasoning and ambiguity and for reducing hallucinations through structured interaction. From a broader coordination lens, Sun et al. [136] presents a unified framework for cooperative behavior, asking who should coordinate with whom and how across diverse tasks. Yan et al. [169] then moves between system-level and internal communication, relating architectures, goals, and protocols to strategies, interaction paradigms, and message content; they surface challenges in efficiency, security, and scalability. Finally, Tran et al. [143] concentrates on collaboration enablers such as coordination strategies and communication structures that help multi-agent systems work effectively.

Several surveys address trust and governance. Raza et al. [122] situate AI TRiSM, trust, risk, and security management—over the full lifecycle, bringing together explainability, secure orchestration, privacy management, and governance to strengthen reliability, robustness, and responsible practice. Yu et al. [175] proposes a modular, security-oriented view of trustworthy LLM agents and MAS, extending trustworthy-LLM notions to agent settings. They decompose systems into intrinsic modules such as language model, memory, tools; extrinsic context—user, other agents, environment—and review attacks, defenses, and evaluations for each, including working definitions of fairness and robustness. From a normative and organizational perspective, Constantinescu and Kaptein [32] analyze accountability in complex sociotechnical chains and introduce the M3 framework of many agents, many levels, and many interactions. They argue that responsibility should concentrate on actors at central interaction nodes with outsized influence, such as model providers, and that obligations should evolve dynamically with changing interaction patterns and regulation. Within single-agent settings, Zhao et al. [190] surveys LLM explainability with a focus on local and global methods, contrasting with our system-level, multi-agent perspective.

To our knowledge, none of these surveys offers a single, end-to-end taxonomy of trustworthy LLM-based multi-agent systems that jointly covers *Reliability*, *Transparency & Understandability*, *Accountability*, and *Fairness & Ethics*, while also providing formal definitions tailored to MAS and a synthesis of concrete methods under each element. We address this gap. Our taxonomy spans four pillars—Reliability, Transparency & Understandability, Accountability, Fairness & Ethics—and ties each pillar to implementable mechanisms in MAS. Under Reliability we organize accuracy, robustness, resilience & recoverability, and connect them to levers such as routing, aggregation rules, debate and critique protocols, control-loop recovery, and runtime enforcement. Under Transparency we distinguish explainability and interpretability, uncertainty and limitations, and reproducibility; we make clear that explainability offers human-readable reasons, whereas interpretability exposes the system's operating policies and flows, including routing choices, stopping thresholds, and leadership signals. Under Accountability we move beyond conceptual discussion to operational tools: agent- and step-level attribution, explicit trace graphs, quantitative influence indices such as Shapley and leave-one-out as well as voting ledgers, and auditable standardized logs and protocols. Under Fairness & Ethics we situate alignment, diversity, and equity within multi-agent workflows, embedding evaluation and mitigation directly into pipelines through auditor agents, consensus repair, and influence moderation.

The closest related effort outside MAS is the FATE survey in higher-education AI [98], which systematically reviews definitions and studies of fairness, accountability, transparency, and ethics in that domain. In contrast, our focus is LLM-based multi-agent settings. Rather than primarily cataloging concepts and calling for more technical study, we provide an operational taxonomy with levers that practitioners can build and evaluate, and we articulate how the four pillars interlock in MAS, including the dependencies between reliability, responsibility, and governance.

C Extended Discussion of Accuracy and Consistency Mechanisms

This section provides additional implementation and evaluation details for the five coordination paradigms discussed in Section 2.1. These details complement the synthesis in the main text but are not required for distinguishing the mechanisms themselves.

Debate-based Coordination. The corrective effect of debate depends on agents having sufficiently different reasoning paths. Du et al. [40] show that multi-agent debate improves factuality and reasoning relative to single-agent and self-reflection baselines when agents hold genuinely divergent views. Self-reflection alone may instead reinforce an agent's initial position, making external challenge an important source of correction [12, 91].

Recent work further separates the benefit of deliberation from the ensemble effect of querying multiple agents. Choi et al. [28] model debate as a martingale process and show that, under their assumptions, iterative communication does not increase expected correctness on average. Their experiments across seven benchmarks similarly find that majority voting accounts for much of MAD's performance, with additional communication sometimes overturning initially correct answers. Fan et al. [47] address this limitation through iMAD, which predicts whether an initial answer is likely to benefit from further deliberation using self-critique and linguistic uncertainty signals. Debate is triggered only when recovery appears probable. On GSM8K, iMAD achieves higher accuracy than always-on MAD while using 70% fewer tokens, and on MEDQA it reduces token use by 92% while outperforming MAD and GroupDebate.

The organization of debate rounds also affects their reliability. Zhang et al. [187] find that speaking order can exert greater influence over the collective decision than answer correctness, with later-positioned agents consistently receiving greater influence. Truth Last configurations outperform Truth First by up to 24 percentage points on some models. Their MADC framework uses path consistency to identify a more reliable viewpoint and places it in the final speaking position. Zhang et al. [185] instead adapt role assignment through Meta-Debate, using proposal and peer-review stages to select models for debate roles according to task-specific suitability. Their DMAD configuration reaches 66.29% accuracy on GPQA and outperforms homogeneous and random role assignments.

Debate has also been applied to evaluation and adversarial settings. Hu et al. [60] use iterative debate among LLM judges and find that revision after observing other judges' reasoning improves evaluation accuracy on more difficult tasks. Adaptive stopping ends most debates within two to eight rounds with little accuracy loss, while on simpler tasks additional discussion can degrade performance when initial agreement is already high. Yang et al. [172] combine adversarial debate, error-logged self-stabilization, and weighted voting to limit hallucination propagation. Finally, Zhang et al. [182] find that homogeneous debate can underperform single-agent baselines in some settings, suggesting that part of the apparent benefit of debate may arise from model heterogeneity rather than interaction alone.

Role-based Coordination. Role-based systems instantiate specialization in different ways. AutoGen uses conversable agents with flexible speaker selection and reusable modules, supporting role-based, human in the loop, and tool augmented configurations within the same framework [161]. Mixture of Agents instead uses a layered architecture in which proposer agents generate independent responses and downstream aggregators synthesize them into a final answer [146]. SMoA makes this architecture more selective by activating only a subset of candidate agents, applying judge-based filtering between layers, and terminating once consensus is reached, matching MoA on several alignment and reasoning tasks with substantially lower token consumption [76].

Model specialization provides another form of role assignment. Ye et al. [174] show that assigning heterogeneous LLMs to roles that better match their individual strengths improves performance on mathematics and reasoning tasks

relative to homogeneous teams. ReConcile similarly uses agents from different model families in a round-table discussion followed by confidence-weighted voting, allowing agents to revise their answers before collective aggregation [22].

Agent Selection and Routing. DyLAN performs agent selection through an Agent Importance Score derived from peer evaluations during multi-round interaction. Lower value agents are deactivated while stronger contributors remain active, and execution stops once more than two thirds of active agents agree [90]. GPTSwarm represents collaboration as a directed acyclic graph in which nodes implement model or functional components and edges determine information flow. It jointly optimizes node behavior and graph structure and applies self-consistency voting to stabilize the resulting outputs [200]. AgentNet removes centralized routing by allowing individual agents to decide whether a task should be executed, forwarded, or divided, while prior successful execution steps are retrieved to support later routing decisions [171].

Task Decomposition. Chain of Agents partitions long documents across worker agents and uses a manager to integrate their outputs, limiting the amount of context processed by each worker and reducing distraction from irrelevant information [189]. MacNet represents decomposition through a directed acyclic graph in which downstream agents receive refinement instructions from upstream agents, allowing progressively improved solutions to propagate through the network [117]. ChatDev applies staged decomposition to software engineering by assigning design, coding, testing, and related responsibilities to different agents, creating intermediate artifacts that can be inspected before the workflow proceeds [116].

Prompt Engineering. Chain-of-Thought (CoT) prompting elicits explicit intermediate reasoning before a final answer is produced, improving correctness across many reasoning tasks [156]. Layered CoT extends this idea by placing verification and interaction agents between reasoning stages so that contradictions can be detected before later steps depend on them [125]. ReAct alternates reasoning with actions in an external environment, allowing retrieved evidence and tool responses to inform subsequent decisions rather than relying only on parametric knowledge [173]. CAMEL uses inception prompting to instantiate persistent user and assistant roles and ground their interaction in explicit goals, enabling autonomous collaboration while reducing drift from assigned roles [77].

D Accuracy & Consistency Failure modes, and targeted remedies

Although the coordination mechanisms discussed above can improve accuracy and consistency, they do not eliminate failures introduced or amplified through multi-agent interaction. Recurring failure modes include hallucination propagation, prompt sensitivity, sycophancy, reasoning degradation over long contexts, and reward hacking. We discuss these failure modes below together with the mechanisms through which they arise and the remedies proposed in the literature.

D.1 Hallucinations in Debate

Agents may hallucinate during debate, and the multi-agent setting introduces a compounding dynamic absent in single-agent contexts: a hallucinated claim asserted confidently by one agent can propagate through subsequent rounds as other agents update on it rather than independently verifying it. Yang et al. [172] address this through a framework that first stabilizes each agent through repetitive inquiry with error logging and then cross-checks answers via adversarial debate and weighted voting, reporting steadily rising composite accuracy and reduced hallucination rates across twenty evaluation batches. ChatDev [116] approaches the same problem through communication scaffold

design, incorporating a Communicative De-hallucination protocol that enforces explicit turn-taking and constrains speculative statements during collaborative reasoning, demonstrating that structured communication can suppress factual drift without additional supervision.

D.2 Prompt Sensitivity

Small, superficial changes to prompts systematically shift model behavior, causing accuracy to vary substantially across near-equivalent phrasings of the same question [57]. In multi-agent settings, this sensitivity is compounded because each agent's output becomes part of the input context for downstream agents, meaning that prompt-induced variation in one agent propagates through the pipeline. Herr et al. [57] measure this effect in two-player non-zero-sum games, finding that near-equivalent prompts produce inconsistent strategic choices across models. CMAT mitigates low-quality and sensitive prompts through three functional roles, User, Assistant, and Checker, combined with short-term and long-term memory and self-reflection for adaptation over time [86]. Checker feedback corrects errors directly, while CoT [156] and ReAct [173] stabilize intermediate reasoning and improve correctness across task types.

D.3 Sycophancy

LLM agents tend to align their outputs with perceived expectations or with the positions of other agents, particularly those assigned higher authority or speaking earlier [187]. In multi-agent settings, this creates a cascade risk distinct from single-agent sycophancy: once one agent defers to another's incorrect position, subsequent agents may follow, producing apparent consensus around a wrong answer. Xie et al. [167] examine this cognitive bias in LLM-based MAS, showing that agents exposed to confident but incorrect peer outputs systematically shift their responses toward those outputs. The positional bias documented by Zhang et al. [187], where later-speaking agents dominate regardless of correctness, is partly a manifestation of this tendency. Mitigation strategies include diversity-preserving mechanisms such as implicit consensus [162], which prevents premature lock-in by having agents commit to actions independently, and heterogeneous model assignment [174], which reduces the likelihood that all agents share the same biases.

D.4 Reasoning Degradation over Long Contexts

As pipelines grow longer and inter-agent message histories accumulate, reasoning quality deteriorates in ways that have no direct single-agent equivalent. In MAS, each agent-to-agent handoff is a compression point at which information is summarized, and compression losses accumulate across stages. Ao et al. [8] formalize this as communication loss, showing that under log loss the degradation at each stage becomes conditional mutual information, and that a five-agent relay falls below random chance on MMLU because successive summarizations distort rather than preserve the decision-relevant information. Qi et al. [115] situate this within a broader taxonomy of MAS failure, observing that long interaction trajectories make failures difficult to diagnose because relevant evidence becomes mixed with redundant,noisy, or propagated information, and that sequential interaction is particularly vulnerable to cascading errors because early mistakes constrain the reasoning available to later agents. At the task execution level, Wan et al. [145] identify two distinct failure patterns that emerge as trajectories lengthen: context overload, in which critical constraints become buried within lengthy histories, and context amnesia, in which earlier evidence and previously failed approaches disappear from the working context, leading to repeated tool errors, constraint drift, hallucinations, and premature termination. Their COMPASS framework addresses both through a three-agent architecture in which a Main Agent performs step-by-step reasoning and tool use, a Meta-Thinker monitors for loops, strategy drift, and premature stopping, and a Context Manager compresses accumulated histories into concise briefs containing verified

evidence, open questions, and next actions. Ablation results illustrate the contribution of each component: removing the Meta-Thinker reduces BrowseComp accuracy from 35.4% to 15.2%, while removing the Context Manager reduces it to 26.4%, confirming that both oversight and active context organization are necessary for sustained reasoning quality. Chain of Agents provides a complementary structural mitigation by bounding each agent's context to a single document segment and passing only a managed state forward [189], while Pezeshkpour et al. [111] formalize a Reasoning Capacity notion that identifies the weakest links in a pipeline under budget constraints and applies targeted repair to those components.

D.5 Reward Hacking

In MAS settings where agents are optimized through feedback, such as debate with LLM judges or reinforcement-learning-tuned coordination, agents may find shortcuts that satisfy the evaluation metric without achieving the underlying objective [17]. This failure mode is particularly relevant when the judge is itself an LLM agent, since the judge's evaluation criteria can be reverse-engineered by sufficiently capable debaters. Hu et al. [60] observe this risk in multi-judge debate, noting that some agent configurations converge on superficially confident but factually weak answers that nonetheless score well under peer review. Process reward models trained on synthetic reasoning data offer one mitigation by evaluating the quality of intermediate reasoning steps rather than final outputs alone [177], reducing the incentive for agents to optimize for terminal answer appearance at the expense of reasoning integrity.

E Extended Discussion of Robustness Mechanisms

This section provides additional empirical and implementation details for the three robustness axes discussed in Section 2.2.

E.1 Non-Adversarial Robustness

Distribution Shift and Partner Variation. Distribution shift includes changes in tasks, tools, environments, and the other agents with which a system must coordinate. Reasoning Capacity estimates the probability of correctness under practical constraints and can identify bottlenecks that limit performance as operating conditions change [111]. Partner variation presents a related challenge. Agashe et al. [2] find that agents can coordinate relatively reliably with previously unseen partners in pure coordination games, but performance deteriorates substantially when tasks require Theory of Mind reasoning or inference over hidden beliefs [78]. This suggests that robustness to partner shift depends on the complexity of the information agents must infer about one another.

Long-Context and Tool Degradation. As multi-agent pipelines grow longer and increasingly depend on external tools, context truncation, retrieval failures, and noisy tool outputs become additional sources of degradation. Chain of Agents limits the context processed by each worker and passes controlled summaries between agents, reducing distraction and retrieval related failures in long-context tasks [189]. This provides an example of improving robustness by constraining information exposure rather than simply increasing context capacity.

Architectural and Workflow-Level Solutions. System-level architecture can provide robustness independently of improvements to the underlying model. Geambasu et al. [50] argue that dynamically constructed agents are structurally brittle because they bypass software-engineering practices such as validation, staged deployment, and maintenance. Their AI Workflow Store instead emphasizes reusable and pre-engineered workflows for high-stakes settings.

Guan et al. [53] provide production-scale evidence for this principle. Their environmental data management system separates data collection, validation, publication, and orchestration responsibilities across agents and places deterministic validation before irreversible publication. In one deployment, a validator detected a coordinate-transformation error affecting more than two thousand stations within ten minutes and prevented it from reaching users, even though the producing agent had failed to recognize the error. The example illustrates how explicit role separation and cross-agent validation can prevent locally plausible failures from propagating into system-level outcomes.

Resource Constraints. Resource robustness concerns maintaining output quality under limits on tokens, latency, and computational cost. SMoA activates only selected agents and applies early stopping after sufficient agreement is reached, reducing resource use while limiting unnecessary low-quality contributions [76]. Mixture of Agents separates proposal generation from aggregation without additional training [146]. On a larger scale, MegaAgent combines on-demand agent spawning, parallel execution, hierarchical scheduling, and agent checklists to limit unnecessary interaction and control cost and latency [149].

E.2 Adversarial Robustness

Prompt Injection and System Compromise. Prompt injection exploits the natural-language interfaces through which agents consume retrieved documents, emails, tool outputs, and other external information. Zhan et al. [178] evaluate this threat across 1,054 test cases and show that prompted agents are highly vulnerable, while fine-tuned tool-use agents exhibit greater resistance. Indirect prompt injection can also propagate across multi-step workflows after a single poisoned retrieval event [52]. At the system level, Tian et al. [141] show that compromising high-authority roles can be particularly damaging in hierarchical systems because malicious instructions propagate to downstream agents.

Debate Manipulation and Strategic Bias. Adversarial manipulation can exploit the coordination mechanism itself. Amayuelas et al. [6] show that a malicious participant can steer multi-agent debate by generating persuasive arguments, with attack effectiveness depending more on persuasiveness than formal authority. Herr et al. [57] identify positional, payoff, and behavioral biases in strategic decision settings and find that CoT prompting can reduce these biases for some models while amplifying them for others. These results caution against assuming that reasoning-oriented interventions uniformly improve robustness.

Defenses. Defense mechanisms intervene at different stages of the attack lifecycle. RedHit iteratively generates and refines adversarial prompts for adaptive stress testing [132]. Randomized smoothing has been adapted to multi-agent consensus by perturbing and verifying neighboring reports and using trimmed-mean aggregation to provide probabilistic robustness guarantees [59]. Other work filters adversarial and low-quality output through auxiliary language models before they enter the shared interaction process [82]. Error-logged self-stabilization and weighted voting provide communication-level defenses against hallucination cascades [172]. FAIRGAME addresses strategic preference and payoff biases through mechanism design and counterfactual self-play across agent configurations, languages, and interaction settings [15].

E.3 Inter-Agent Robustness

Malicious and Insider Agents. Failures originating from within the agent population can be difficult to distinguish from ordinary variation. Sun et al. [137] train an attacker within a learned model of benign agent dynamics and show

that inferred behavioral characteristics can be exploited to sustain disagreement. In their experiments, consensus falls from 0.95 to 0.83 while the attacker remains difficult to distinguish from a cooperative participant. The finding is particularly important because inferred behavioral attributes are nearly as useful as directly known ones, meaning that effective insider attacks do not necessarily require privileged system access.

Xie et al. [166] study intention-hiding agents that use strategies including suboptimal fixation, reframing, fabricated evidence, and strategic delay. Their behavioral screening approach improves team performance when flagged agents are down-weighted, with layered communication structures showing comparatively strong robustness. These studies suggest that inter-agent robustness requires monitoring behavioral consistency across an interaction trajectory rather than inspecting isolated outputs.

Topology and Communication Structure. Topology determines both how quickly an error spreads and whether the system can route around degraded components. Chen et al. [23] compare four multi-agent topologies under dark personality agent injection and find that shared-pool architectures are especially vulnerable, decentralized configurations are more resilient, and repeated debate can amplify risk. Qian et al. [117] similarly find that increasing team size improves accuracy and stability only when accompanied by explicit voting and selection mechanisms.

AgentNet addresses structural fragility through decentralized routing in which agents locally decide whether to forward, divide, or execute a task, eliminating dependence on a single central planner [171]. GPTSwarm instead learns sparse collaboration graphs that remove low-value communication links, reducing both noisy information propagation and unnecessary token expenditure [200].

Aggregation Under Stress. Aggregation determines how local uncertainty and error are converted into a collective decision. Increasing the number of agents can improve accuracy and stability when paired with majority or weighted voting, while early stopping limits the opportunity for random errors to influence subsequent interaction [81]. Implicit consensus instead allows agents to exchange reasoning while maintaining independent actions, reducing premature convergence and preserving alternative trajectories [162].

Reflective collaboration introduces reviewer and critique roles that can stabilize intermediate outputs, although these roles may inherit errors or biases present in earlier drafts when no independent challenge is available [12]. Risk-aware decision policies incorporate calibrated confidence, deferral, and escalation to reduce the effect of noisy inputs and unreliable tool responses [74]. ReSo further adapts the collaboration structure by pruning low-value assignments under changing task and budget conditions [193].

Architectural Mechanisms. System-level architectural mechanisms provide another layer of inter-agent robustness. BlockAgents uses tamper-evident decision records and incentive-based coordination to discourage Byzantine behavior, collusion, and message manipulation [19]. In open agent societies, Piatti et al. [113] show a different form of fragility. The introduction of one greedy participant can increase inequality, destabilize cooperation, and trigger collective collapse, with negotiation emerging as a key mechanism for restoring cooperation.

MegaAgent combines topology awareness, aggregation control, monitoring, rerouting, and resource management within a hierarchical architecture [149]. Its admin-worker structure localizes failures within subtask groups and allows monitors to reroute or escalate problematic decisions, illustrating how multiple robustness mechanisms can be combined within the same large-scale system.

F Extended Discussion of Resilience and Recoverability Mechanisms

F.1 Rollback, Replay, and State Restoration

Versioned artifacts and execution logs make it possible to resume a pipeline from a known good state rather than restarting from scratch, which is particularly important when failures occur late in a long workflow. Mazaheri [96] operationalize this through REPOT, which deterministically replays a generated plan to identify its longest valid prefix, preserves that state as a checkpoint, and uses a single additional LLM call to repair only the remaining suffix. The checkpoint proves decisive: checkpoint-aware methods achieve over 70% recovery on Gemini compared with at most 3.1% for error-only feedback.

At scale, Wang et al. [149] achieve state restoration through explicit agent states, queue-based orchestration, and multi-level monitors that restart interrupted subtasks and regroup teams as workload evolves. Zhang et al. [188] complement this by attributing failures to specific agents and time steps, enabling targeted reruns without discarding the work of unaffected agents. Li et al. [82] add a non-parametric safety rollback that shifts the system toward a safer operating regime without retraining when outputs drift toward harmful or low-quality behavior.

F.2 Corrective Loops and Memory

Yang et al. [172] anchor corrective feedback through error-logged self-stabilization, where detected divergence triggers corrective reprompts. Huang et al. [63] add a Challenger gate at the agent level and an Inspector reviewer globally, catching faulty messages before they propagate and restoring up to 96.4% of performance lost under injected faults. Bo et al. [12] similarly pair explicit reviewer feedback with revision actions so that flawed earlier drafts are corrected rather than compounded.

At the individual agent level, Su et al. [134] show that structured reflection can be trained rather than merely prompted. Agents learn to diagnose failed tool calls and generate corrected actions through a reinforcement learning objective, improving long-context tool-call accuracy by 47-56% across models and outperforming both prompt-only reflection and supervised fine-tuning.

At the memory layer, Liang et al. [86] show that short-term and long-term memory with self-reflection can turn isolated failures into repairable events. Reliable memory-based recovery additionally requires source attribution so that agents can distinguish verified prior outcomes from hallucinated inferences [154].

Shen et al. [129] extend correction to the multi-agent level through MASC, which compares each step against a learned prototype of normal behavior and routes anomalous outputs to a correction agent before they enter the shared interaction history. A single faulty agent reduces system performance by up to 51.9 percentage points in their experiments, illustrating the cost of unchecked propagation.

Tamang and Bora [138] complement this with Enforcement Agents that continuously monitor peer messages and intervene through quarantine, rerouting, or escalation while producing traces for later analysis. Nanda et al. [103] provide production-scale validation through Wink, an asynchronous observer for coding agents that periodically injects course-correction guidance without blocking the main execution loop. Wink achieves a 90.93% recovery rate across more than 10,000 intervention-triggered trajectories and reduces the measured misbehavior rate from 18.61% to 15.14% in a live A/B test.

F.3 Consensus Repair and Team Adaptation

A wrong or unstable collective decision does not always require starting over. Liu et al. [90] use mid-run roster adaptation to remove agents whose contributions are degrading the collective trajectory and adjust the stopping rule so that the system can pivot without a complete restart.

Wu and Ito [162] identify premature convergence as a recovery risk. Their implicit consensus mechanism allows agents to share reasoning while committing to actions independently, preserving enough diversity for the group to retain a path back to a correct answer after an early wrong turn. Piatti et al. [113] show that open communication can similarly support partial recovery of cooperation after a disruption to established group norms.

For settings with several plausible trajectories, Chen et al. [22] use round-table re-aggregation to recombine weaker trajectories with stronger ones, while Li et al. [81] use resampling and hierarchical voting to redirect the group after unsuccessful initial attempts. Chen et al. [19] use multi-round re-election when consensus fails and record proposals in tamper-evident logs. Chen et al. [20] extend this through AgentChain, where stake-based incentives favor higher-quality agents while low-quality or malicious contributors are progressively excluded.

F.4 Agent-level Routing and Structure Reconfiguration

Yang et al. [171] remove the central orchestrator and allow agents to route tasks over a directed acyclic graph, reinforce successful edges, and prune low-value links so that the network reshapes itself around successful execution. Zhuge et al. [200] similarly edit collaboration edges so that degraded communication channels can be bypassed.

Guo et al. [54] extend dynamic graph repair through DynaGraph, which monitors confidence and uncertainty during execution and triggers either fine-grained patching by inserting an auxiliary node or subgraph reconstruction by replacing an entire corrupted reasoning branch. Removing subgraph reconstruction reduces MATH accuracy from 82.7% to 41.5%, while the full system reduces token consumption and latency by roughly 65% relative to Reflexion.

Zhou et al. [193] track agent reputations over time and restructure task assignments when poor performance persists. Qian et al. [117] document controlled degradation as teams grow beyond their effective size, showing that team structure itself can buffer cascading failure. In safety-critical settings, Chen et al. [23] combine topology-aware isolation of risky agents with rerouting to restore safe operation without dismantling the broader team.

Li et al. [76] address resource-constrained recovery by activating only top candidate agents and stopping once consensus is reached, limiting the entry of low-quality contributions. Wang et al. [154] situate these mechanisms within a broader recovery framework, noting that topology-level repair is most effective when execution provenance identifies which edges or roles contributed to the failure, allowing reconfiguration to be targeted rather than exploratory.

F.5 Extended Discussion of Attribution Methods

F.5.1 Structural Attribution. The most direct path to attribution is architectural. If every action, artifact, and state transition is linked to an identifiable agent at the moment it is produced, attribution evidence does not need to be reconstructed entirely after the fact. AutoGen [161] logs every utterance, tool call, and termination reason against a named role, yielding provenance through the full interaction history. ChatDev [116] achieves the same through a role-separated pipeline in which distinct agents own distinct artifacts at each stage. CMAT [86] extends this to the feedback level by separating the User-Assistant-Checker loop so that every correction remains tied to the agent that issued it.

At larger scales, MegaAgent [149] binds artifacts and state transitions to concrete agents through checklists and boss-level reviews, while ReSo [193] maintains a Dynamic Agent Database that accumulates attribution records across runs rather than allowing them to expire at the end of a session.

In production settings, Guan et al. [53] require agents to attach **provenance metadata** before inter-agent handoff, treating each transition as a trust boundary at which the source of information and actions is recorded prospectively. Their deployment detected a coordinate-transformation error affecting 2,452 stations within ten minutes with zero user exposure. The detection itself results from validation rather than attribution, but the associated provenance illustrates how explicit handoff records can support later identification of the producing agent and affected stage. Wang et al. [154] formalize this broader idea through execution provenance, a typed causal graph connecting outputs to inputs, tool calls, memory operations, and agent messages.

F.5.2 Marginal and Counterfactual Attribution. Structural logging records what happened and who acted, but does not quantify how much each contribution shaped the collective outcome. The **Shapley value** [128] distributes a collective outcome among contributors according to their marginal contribution, while Mao and Gratch [94] extend related causal reasoning to multi-agent settings by modeling how causal relationships among agents determine individual responsibility for joint outcomes. Pezeshkpour et al. [111] apply Shapley-style analysis to detect dominant agents whose contributions suppress teammates' reasoning, while Bo et al. [12] operationalize this through **counterfactual rewards**, marginalizing each agent's reflection in turn and using the resulting signal to drive targeted correction. SHARP [84] extends this principle to training by estimating planner and worker marginal contributions through counterfactual masking and using agent-specific rewards to reduce harmful subagent calls and improve coordination. DAG-aware memoization reduces the required coalition evaluations by 83.7%.

Cui et al. [33] take a complementary efficiency approach through a **leave-one-out approximation**. After deliberation, peers re-evaluate the outcome while excluding a specified agent, isolating that agent's marginal influence while reducing computational cost from $O(TN^2)$ re-debates to only $O(N)$ additional prompts. HiveMind [165] closes the loop between measurement and improvement through DAG-Shapley, using contribution scores to identify underperforming agents and refine their prompts during operation.

For failure-specific counterfactual attribution, CAR [127] models trajectories as **structural causal models** and estimates each step's causal effect through repeated replay interventions. It distinguishes the step that decided on a harmful action from the one that merely executed it and applies a Monte Carlo Shapley estimator to divide responsibility among interacting steps. CAR is validated primarily on single-agent trajectories, making its extension to multi-agent pipelines a promising direction rather than an established result.

F.5.3 Semantic and Content-Based Attribution. SLIC [69] formalizes semantic attribution through **Semantic Cooperative Games**, tracing how agents generate, preserve, combine, or transform task-relevant semantic information. Its **Semantic Shapley Value** assigns credit according to informational contribution rather than structural position. The method reports 93.3% lower computational cost than exhaustive Shapley evaluation while maintaining strong alignment with human judgments.

A useful consequence of this formulation is the separation between semantic contribution and structural privilege. A final-stage agent may introduce little new information while retaining substantial power to change or damage the final output. Ebrahimi et al. [42] approach a related problem through a **Contribution Score** that estimates each agent's marginal impact on the overall outcome and peer persuasion dynamics, together with a **Credibility Score** that captures

how much influence an agent should receive during aggregation. Together these measures provide an interpretable attribution profile that can directly inform consensus weighting.

F.5.4 Behavioral and Influence Attribution. MADC [187] finds that **speaking order** can exert stronger influence over collective decisions than answer correctness, with later-positioned agents consistently dominating final outcomes regardless of contribution quality. The method responds by reordering agents according to path consistency, making influence assignment an explicit design choice. Kasprova et al. [71] use **pairwise influence matrices** to show how sycophancy propagates across multi-agent discussions and amplifies errors through the team even when no single agent originates them. Providing agents with peer sycophancy rankings reorganizes influence toward more reliable contributors and improves majority accuracy by 10.5 percentage points. BlockAgents [19] implement influence tracking within the operational framework through a **Proof-of-Thought consensus model**. Per-round contribution scores accumulate into an evolving influence index that can expose dominance, bias, or collusion. DyLAN [90] derives a continuous influence measure from task relevance, reasoning confidence, and historical performance and uses it to govern agent activation and output propagation within an adaptive collaboration graph. Because this estimate depends partly on self and peer assessment rather than causal inference, confidently erroneous agents may receive excessive influence through recursive feedback. Shapley-Coop [61] extends influence attribution to settings with conflicting agent goals, using pre-action Shapley reasoning to negotiate compensation and post-task redistribution to align incentives with actual contribution. The method achieves full cooperation in settings where unstructured negotiation produces only 25%.

Finally, Sun et al. [137] reveal an **adversarial dimension** in which a malicious insider learns a model of benign agent dynamics, infers peer behavioral tendencies from observed trajectories, and exploits them to prevent consensus while appearing cooperative. This demonstrates that the same interaction structure that enables influence attribution can also create an attack surface.

F.5.5 Failure Attribution and Benchmarks. Zhang et al. [188] define the **decisive error** as the agent-step pair (i^*, t^*) whose correction changes a failed execution into a successful one. Across 127 systems, they report agent-level attribution accuracy of approximately 53.5% but step-level accuracy of only 14.2%, illustrating the difficulty of identifying the precise point at which failure becomes decisive.

TraceElephant [24] shows that **trace completeness** is itself a first-order determinant of attribution quality. Full observability over inputs, tool interactions, and environment responses improves agent-level accuracy by 22% and step-level accuracy by 76% relative to output-only logs. The evaluation covers spectrum-based fault localization, dynamic replay, and counterfactual approaches, with tool-interacting agents and orchestrators accounting for a substantial share of failures. RAFFLES [196] uses a **Judge-Evaluator architecture** in which specialized evaluators independently assess whether a candidate step is an actual fault, the earliest relevant fault, and decisive for the final outcome. Iterative critique is stored in memory for refinement. On Zhang et al. [188], the method exceeds the strongest simulation-free methods by more than 13 percentage points on hand-crafted trajectories. AGENTRX [10] takes a **constraint-based approach**, synthesizing global and dynamic constraints from tool schemas and the evolving trajectory. The constraints are checked programmatically at each step to produce an evidence-linked violation record before assigning one of nine root-cause categories, improving step localization by up to 23.6 percentage points. StepFinder [198] addresses step-level precision through a lightweight neural model that encodes acting-agent identity together with step content, models temporal evolution through a BiLSTM, and scores steps by failure likelihood in under one second per trajectory, substantially reducing the human debugging search space.

For graph-structured agent dependencies, GraphTracer [183] constructs **causal information dependency graphs** from execution logs so that failure paths can be traced through graph structure rather than flat log inspection. AgenTracer [181] instead trains a lightweight attribution model through **multi-granular reinforcement learning** to jointly predict responsible agents and steps.

A fundamental challenge running through these methods is that failure attribution may be inherently **ambiguous**. MP-Bench [65] finds that only 16.2% of annotated failure steps were selected by all three expert annotators, with pairwise disagreement reaching approximately 60%. This suggests that evaluation frameworks may need to reward multiple plausible and well-supported attributions rather than requiring agreement with one fixed label. Attribution becomes harder still when failure is **intentional**. AgentXposed [166] studies agents that appear cooperative while covertly sabotaging the team through suboptimal fixation, reframing, fabricated evidence, and strategic delay. Progressive behavioral monitoring is used to identify likely culprit agents even when individual outputs appear benign in isolation.

Finally, Qi et al. [115] situate these efforts within a broader lifecycle analysis, observing that attribution becomes less reliable as trajectories grow because information redundancy weakens long-range causal dependencies. They identify the incomplete **attribution-verification-repair loop** as a central open problem, since existing systems frequently diagnose failures without converting those diagnoses into structural changes that prevent recurrence.

F.6 Extended Discussion of Traceability Mechanisms

F.6.1 Execution Artifacts and Structured Observability. Span based observability approaches capture planning, task execution, model invocation, tool use, evaluation, guardrail activity, and external interactions within a shared trace hierarchy. AgentOps [39] develops a taxonomy of the artifacts and metadata that should be recorded across the agent lifecycle, including timestamps, trace and parent identifiers, model parameters, inputs and outputs, retrieved knowledge, task dependencies, errors, evaluations, and intervention events. AgentTrace [5] provides a complementary runtime schema that organizes such records across cognitive, operational, and contextual surfaces. Together, these approaches connect prompts, generated plans, method calls, tool results, database or filesystem access, execution errors, and environmental effects within a temporally ordered execution record.

Task specific frameworks retain more specialized intermediate artifacts. Chain of Agents records worker outputs, supporting source spans, and synthesis information [189], while Layered Chain-of-Thought preserves layer specific intermediate results, generated rationales, and verification outcomes [125]. These records connect final outputs to intermediate claims, evidence, and checks, but should not be interpreted as complete or necessarily faithful access to the underlying model computation.

F.6.2 Inter-Agent Communication and Task Flow. Dialogue based systems provide a basic form of communication lineage by preserving role conditioned exchanges, speaker order, and contextual responses. CAMEL [77] records chronological role based interaction, allowing a collaborative trajectory to be inspected turn by turn. Interactional Fairness [11] supplements dialogue history with Critical-Incident tags and an Explanation Journal recording the justification, tone, and context associated with fairness relevant interactions [30].

AutoGen [161] provides a more structured execution record through typed events that may include role prompts, speaker selection decisions, messages, tool or function calls with arguments and return values, human interventions, context updates, and termination reasons. These records make it possible to determine which agent initiated an interaction, what information the recipient received, whether an intermediary transformed the content, and what subsequent actions followed. Preserving message identities, sender and receiver roles, delegation links, ordering, and

message transformations is particularly important when reconstructing failures that cross several agent boundaries [72, 161].

F.6.3 Structural Provenance and Information Flow Tracing. GPTSwarm [200] represents an LLM-MAS workflow as a directed acyclic graph of model and tool calls. Recording node inputs, outputs, prompts, and retrieved context yields an explicit provenance structure over the workflow topology. PROV-AGENT[133] provides a broader provenance representation by extending W3C PROV with agent specific entities and activities, including agents, prompts, responses, model invocations, tools, decisions, telemetry, workflow tasks, and domain data. Standardized relations connect these elements and support backward tracing from a decision to its inputs as well as forward tracing from an earlier decision or artifact to later tasks and outputs.

Information flow tracing follows particular content across transformations, tools, memory, sessions, and agents. NeuroTaint [16] constructs a dynamic context provenance graph that tracks untrusted information from source events to privileged sinks. It distinguishes direct reuse, paraphrased or fragmented propagation, implicit control influence, and delayed reuse through persistent memory. This allows the trace to capture cases in which earlier information affects a later action without appearing as direct textual reuse.

F.6.4 Post-hoc Failure Backtracking. Who&When [188] formalizes failure localization at both the agent and step levels by asking which agent introduced the decisive failure and at what point it occurred. TraceElephant [24] extends this evaluation setting with fuller execution observability, recording task instructions, system configurations, step level inputs and outputs, tool interactions, and environmental states. Relative to output only traces, this additional context improves agent level localization accuracy by 22% and step level accuracy by 76%, demonstrating that incomplete records can conceal the origin of failure.

Different localization methods consume these traces in different ways. StepFinder [198] encodes agent identity and execution content as temporal semantic representations and models sequential development and longer range dependencies to rank likely root cause steps. RAFFLES [196] uses a Judge and specialized Evaluators to determine whether a candidate is a genuine fault, whether it is the earliest relevant fault, and whether it decisively changed the trajectory. Evaluator critiques and confidence estimates are retained across iterations so that earlier judgments can be revised.

AGENTRX[10] instead derives execution constraints from tool schemas, domain policies, and the evolving trajectory. It produces evidence linked violation records and uses them to identify the first unrecoverable failure and assign a root cause category. Although these approaches use different localization mechanisms, all depend on trace evidence that connects individual actions to their surrounding execution context.

Failure reconstruction can remain ambiguous even with detailed traces. MP-Bench [65] reports that only 16.2% of annotated failure steps were selected by all three expert annotators, while 56.1% were identified by only one. These results show that deeper traceability does not guarantee one unique attribution, but can provide the evidence needed to compare several plausible reconstructions.

F.6.5 Runtime Monitoring and Provenance-Aware Intervention. Who's the Mole [166] monitors behavioral changes across interaction rounds and uses targeted questioning to expose agents that hide malicious intent, producing temporal evidence of when and under what conditions an agent deviated. Enforcement Agents [138] attach policy check outcomes and violation identifiers to messages and actions and record responses such as quarantine, rerouting, or mandatory review. These records connect an intervention with the triggering action, applicable rule, agent, and local execution

context. Agent-Sentry [126] extends provenance to tool execution by recording where values used in tool calls originate and how retrieved information contributes to sensitive actions. These records are compared with trusted sources and observed execution patterns when deciding whether an action should be allowed, blocked, or escalated. AUTHGRAPH [147] separates realized execution from intended authorization through two related graph representations. One records the actual trajectory, including observations, tool calls, and parameter sources, while the other represents the tools and information flows authorized by the original user request. Comparing the two allows unauthorized tool use or parameter substitution to be detected even when generated model outputs have been influenced by malicious content.

Agent-Sentry and AUTHGRAPH therefore both use provenance as a runtime control signal, but differ in how the relevant boundary is established. Agent-Sentry derives structural boundaries from observed execution, whereas AUTHGRAPH compares realized actions with a separately generated authorization specification.

F.7 Extended Discussion of Auditability Mechanisms

F.7.1 Evidence Capture and Visibility. Visibility mechanisms establish which agents and responsible actors can be associated with consequential behavior. Agent identifiers, activity logs, real-time monitoring, and agent cards can document a deployed agent's goals, permissions, tools, interactions, and responsible stakeholders [18].

Agent frameworks provide more detailed operational evidence. AutoGen records role-scoped conversations, speaker selection, tool interactions, and termination conditions [161]. MegaAgent preserves monitoring and validation results across hierarchical execution [149]. SMoA exposes routing, filtering, and stopping decisions [76], while AgentNet records task dependencies and agent-specific execution fragments [171]. These records show what agents received, produced, selected, rejected, or transferred and thereby support action recoverability, lifecycle coverage, and later attribution analysis.

F.7.2 Audit Protocols and Independent Evaluation. FACT-AUDIT [87] specifies evaluator roles, graded evidence, an adaptive review process, and quantitative measures for assessing factuality and rationale quality. Machine-checkable tests provide a complementary approach by evaluating structural requirements, role boundaries, required output formats, and confidentiality constraints across multi-agent interactions [49]. These mechanisms improve policy checkability by translating general expectations into repeatable conditions that can be applied to individual agents and interaction turns. HarnessAudit [89] extends this principle to the full execution trajectory. Rather than evaluating only the final output, it treats the execution harness, including its allocation of tools, resources, permissions, and communication channels, as the unit of evaluation. It assesses boundary compliance, execution fidelity, and stability under perturbation by examining tool calls, resource access, messages, and state changes across both single-agent and multi-agent executions. Its results show that successful task completion can coexist with violations of permission, resource, or information-flow boundaries.

Human in the loop approaches such as LLMAuditor [7] complement automated checks when the relevant criteria require contextual interpretation or qualitative judgment. Such frameworks preserve structured evidence for external reviewers while allowing human assessment where policy requirements cannot be reduced to deterministic tests.

F.7.3 Continuous Auditing and Intervention. Constraint State Governance [83] represents safety requirements as explicit, versioned constraints and checks authority, action scope, information-flow permissions, and supporting evidence before admitting critical actions. It records the active constraint, proposed action, delegated capability, decision, and justification in a hash-linked audit trail. This allows policy checks and interventions to remain connected to the evidence that triggered them and reduces the risk that constraints are weakened during delegation, memory use, communication,

or tool execution. AgentForesight [179] monitors trajectory prefixes and predicts the suspected decisive step and responsible agent before the error propagates. Enforcement Agents [138] and MedSentry [23] record staged responses such as screening, verification, isolation, and escalation, linking each intervention to the behavior that triggered it. EnviSmart [53] applies similar principles in a production workflow by separating worker, validation, publication, and orchestration roles, preserving intermediate artifacts, and requiring deterministic validation and approval before irreversible actions. These mechanisms illustrate how audit evidence can be used prospectively to prevent or contain failure rather than only retrospectively to explain it.

F.7.4 Post-hoc Evidence for Audit. Post-hoc methods use execution records to identify responsible agents, decisive steps, and underlying violations. Who&When [**?**], AGENTRX [10], and Who's the Mole [166] therefore provide evidence that can support later audits, although their primary methodological contribution concerns failure attribution rather than audit protocol design. Their detailed attribution mechanisms are discussed in Section 3.1.

F.7.5 Evidence Integrity and Controlled Access. Logs, evaluation results, intervention decisions, and incident findings cannot support a defensible audit if they can be altered, selectively removed, or fabricated. Versioning, append-only storage, cryptographic signatures, and hash-linked records can help preserve evidence integrity, while access controls, redaction, and retention policies limit unnecessary exposure of private prompts, retrieved documents, tool parameters, credentials, and memory contents [1, 3, 83, 105].

These requirements are particularly important in LLM-based MAS because audit records may contain information originating from several agents, tools, users, and external sources. The value of an audit therefore depends not only on the amount of evidence collected, but also on whether that evidence remains authentic, appropriately protected, and available to the reviewers authorized to examine it.

F.8 Extended Discussion of Explainability and Interpretability Methods

F.8.1 Grounded and Trace-Based Explanations. Trace-based explanation approaches derive their explanations from artifacts produced during execution rather than relying only on an agent's retrospective description of its own behavior. Relevant evidence can include intermediate outputs, agent states, messages, tool interactions, contextual information, confidence estimates, and decision records. Agentic interpretability work describes this as trace-based interpretability, where interaction traces provide evidence from which the development of a collective decision can be reconstructed [107].

Related work argues more broadly that system-level interpretation requires connecting information across reasoning, planning, memory, actions, and interactions rather than examining these components independently [197]. Layered Chain-of-Thought provides a related form of procedural explanation by dividing reasoning into intermediate stages that can be explicitly verified before later stages proceed, making the resulting process more structured and inspectable [125].

Prototype-based approaches compare intermediate and final behaviors with validated examples and use these comparisons to explain behavior at the agent, interaction, and system levels [101]. These methods reinforce the distinction between traceability and explainability: execution records preserve evidence of what occurred, while explanation mechanisms select, organize, and communicate that evidence in a form intended to support understanding.

F.8.2 Interaction- and System-Level Interpretation. SwarmProbe [195] analyzes how agent stances evolve across interaction rounds and exposes patterns including consensus formation, polarization, leader emergence, and turning points

in collective decision making. Its influence measure captures temporal association between messages and later stance changes rather than establishing causal responsibility. In its user study, providing these system-level explanations increased analysis accuracy from 58.0% to 82.5% while reducing average analysis time from 18.7 to 8.3 minutes.

Yan et al. [170] take a complementary data centric approach to interpreting multi-agent behavior during training. Their framework combines sparse autoencoder features with trajectory summarization and groups low-level features into higher-level behavioral concepts representing strategies and interaction patterns. Across LLM-based multi-agent reinforcement learning trajectories, 90% of the resulting Meta-Feature hypotheses were predictively significant, compared with substantially lower rates for individual features and direct trajectory summaries.

The method also identifies divergence between successful and failed training runs before the difference becomes visible in aggregate reward curves and reveals reward-hacking behavior during training. These results illustrate how system-level interpretability can identify behavioral structure that is not apparent from isolated agent outputs or aggregate task performance.

F.8.3 Fine-Grained Explanations of Agent Behavior. XG-Guard [109] detects compromised agents using sentence- and token-level representations of multi-agent communication and provides token-level explanations identifying which parts of a message contributed to the anomaly score. Because these scores are derived from the anomaly-detection mechanism itself, they explain the basis of the detector's judgment rather than serving as independently generated natural-language rationales. The scope of these explanations remains local. They indicate why one agent was flagged relative to the current dialogue context but do not reconstruct how the complete interaction trajectory produced a later collective outcome. Fine-grained explanation and system-level interpretation should therefore be viewed as complementary rather than interchangeable.

F.8.4 Interactive and Human-Centered Explanations. Interactive interfaces allow stakeholders to request information at a level suited to their expertise and purpose. LLMCheckup [148] demonstrates this approach for individual LLMs by combining feature attribution, rationalization, counterfactual generation, and similarity analysis within a conversational interface. XAgen [152] extends the human-centered perspective to multi-agent workflows. It transforms raw execution logs into an interactive workflow representation exposing agents, tasks, tools, intermediate outputs, and execution order, while allowing users to inspect detected failures and their associated explanations. These approaches replace a fixed explanation artifact with an exploratory process in which users can move between summary and detailed views according to their information needs.

F.8.5 Faithfulness and Explanation Evaluation. Faithfulness is particularly challenging in LLM-MAS because information may be communicated, summarized, transformed, stored, retrieved, or acted upon by several agents before affecting the final outcome. Natural-language rationales may therefore omit or misrepresent relevant dependencies even when they appear coherent. Feature-attribution methods can identify influential signals without fully explaining the decision process, while system-level summaries can hide dependencies among agents, tools, memory, and environmental feedback [93, 102, 197].

One form of evaluation tests whether the elements identified by an explanation are behaviorally consequential. Agents, messages, features, or interaction steps identified as important can be perturbed or removed and the resulting change in system behavior measured. Human-centered evaluation instead examines whether intended users understand and can use the explanation, using measures such as analysis accuracy, error-identification accuracy, task completion time, confidence, and perceived interpretability. Application-level evaluation examines whether an explanation improves

the practical activity it is intended to support, including debugging, supervision, failure localization, intervention, or system refinement.

Recent LLM-MAS work illustrates the need for these complementary measures. SwarmProbe evaluates system-level explanations through users' analysis accuracy, completion time, and confidence when identifying collective behavior [195]. XAgen evaluates whether workflow visualizations and error explanations improve users' understanding and diagnosis of multi-agent executions [152]. Yan et al. [170] combine human ratings of interpretability with behavioral validation, testing whether discovered hypotheses help users distinguish training stages and whether injecting those hypotheses back into an agent changes downstream performance.

F.9 Extended Discussion of Uncertainty and Limitation Mechanisms

F.9.1 Uncertainty Estimation and Representation. Collaborative Entropy (CoE) [135] separates uncertainty arising within individual models from disagreement across models by combining semantic entropy with distributional differences among their responses. This distinction allows the system to separate shared ambiguity from cases in which individually confident models support conflicting answers. DISCOUQ [67] similarly demonstrates that vote counts alone provide an incomplete estimate of collective confidence. It derives uncertainty from properties of the disagreement itself, including the evidence and arguments associated with majority and minority positions and the geometry of agents' reasoning representations. Its linguistic variant reports an average Expected Calibration Error of 0.036 compared with 0.084 for vote-count confidence, indicating closer correspondence between estimated confidence and observed correctness.

The reliability of agreement becomes more difficult to assess after agents have communicated because their errors are no longer independent. CAGE-CAL [62] estimates this effect by comparing an interacting panel with an independent-response reference constructed from the same agents. The authors distinguish *diversity-induced under-confidence* from *communication-induced over-confidence*. The latter occurs when interaction causes agents to converge on the same incorrect answer, making agreement overstate reliability. CAGE-CAL improves mean AUROC from 73.46 to 83.61 over the reported DISCOUQ baseline. MATU [25] characterizes uncertainty through variability across the collaborative trajectory rather than the final answer alone. It embeds intermediate responses from multiple agents and repeated executions into a tensor representation and uses reconstruction error to measure how strongly a run departs from recurring latent reasoning patterns. This allows uncertainty to reflect instability across agents, reasoning steps, communication structures, and repeated executions.

Related multi-model work provides further evidence that individual confidence and collective uncertainty should not be conflated. MUSE [73] separates within-model entropy from disagreement between heterogeneous LLMs and uses these quantities to determine which model subsets contribute to the final prediction. Although evaluated as a multi-LLM ensemble rather than an interactive LLM-MAS, the method shows that diversity is beneficial only when disagreement reflects complementary information rather than noisy predictions. Feng et al. [48] examine uncertainty by probing the same black-box LLM through alternative formulations of a query. Multiple agent instances access the same underlying knowledge from different contexts, exposing variation that repeated responses to the original query may fail to reveal.

F.9.2 Uncertainty Propagation and Evolution. Xia et al. [164] shift attention from initial uncertainty estimation to the effects of uncertainty on later stages of an LLM-based workflow. Their framework emphasizes downstream reuse. When information crosses component boundaries, associated uncertainty may be reformulated, stored, consumed by a control mechanism, or communicated to another agent. Handoffs, control decisions, and persistent state are therefore important

points at which uncertainty may change in meaning or consequence. Zhang et al. [186] similarly propose maintaining explicit uncertainty records throughout execution. Their framework updates information concerning the source of an uncertainty, its supporting evidence, confidence, risk, and dependencies. Uncertainty progresses through lifecycle states including detection, characterization, mitigation, resolution, expiration, and escalation, allowing the system to represent how uncertainty changes rather than treating it as a static score. Emde et al. [46] empirically demonstrate the loss of uncertainty across agent boundaries through a phenomenon they call *vanishing uncertainty*. After information passes from one agent to another, the receiving agent may relay it with substantially less visible uncertainty than was present at the source. In experiments with two agents, likelihood-based uncertainty estimates show weak correlations between the subagent and orchestrator, with $|\rho| \leq 0.35$. One reason is that the receiving model's token probabilities represent confidence in generating its own response rather than the reliability of information inherited from another agent.

Uncertainty can also evolve through repeated interaction. Zhao et al. [191] analyze entropy across agents and interaction rounds and find that early entropy and inter-agent dispersion are strongly associated with later collaboration outcomes. Additional rounds do not necessarily resolve this uncertainty, and unresolved disagreement may persist or become more consequential as interaction continues. Their experiments further show that single-agent execution achieves the highest accuracy in 43.3% of the evaluated settings, demonstrating that additional collaboration does not inherently reduce uncertainty or improve reliability.

A related phenomenon occurs in collective hallucinations. Jamshidi [66] show that repeated peer adoption can cause an unsupported claim to become increasingly dominant while agents become more confident in an incorrect consensus. Their model treats claim propagation as an interaction-dependent process and shows that network topology affects whether hallucinations diminish or spread. Agreement and confidence may therefore increase even when epistemic uncertainty has not actually been resolved.

F.9.3 Uncertainty-Aware Response and Limitation Awareness. The framework of Zhang et al. [186] links changing uncertainty states to possible adaptations including specialist or critic agents, further verification, modifications to interaction structure, restricted autonomy, and human intervention. These mechanisms are presented primarily as part of a system engineering framework rather than as a quantitatively validated adaptation policy. Edwards and Schuster [44] study uncertainty caused by incomplete or underspecified instructions. Their multi-agent configuration separates task execution from ambiguity detection. A specialized Intent Agent monitors the Main Agent's trajectory and can require clarification when important information is missing. The multi-agent configuration reaches a 69.40% task-resolution rate compared with 54.80% when underspecified tasks are executed without clarification. This approach treats uncertainty behaviorally by asking not only how uncertain the agent is, but whether it recognizes when additional information is required. iMAD [47] uses uncertainty-related signals to predict whether an initial answer should be retained or passed to multi-agent debate. This avoids assuming that debate is always beneficial because additional interaction may sometimes overturn an initially correct response. Across the reported experiments, iMAD reduces token use by up to 92% while improving accuracy by up to 13.5%.

Other methods use uncertainty after collaboration has begun to control how agents respond to one another. Duan and Wang [41] use uncertainty estimates to modulate cross-agent attention, assigning different weights to peer responses according to estimated confidence during later debate rounds. Zhu et al. [199] incorporate confidence into both sides of the interaction by training agents to report informative confidence estimates and to condition subsequent belief

revision on the confidence associated with peer responses. These approaches use uncertainty as a coordination signal rather than only as information reported after a decision.

Uncertainty may also guide abstention. In DIVERSEAGENTENTROPY, variation across agent instances exposed to different query contexts is converted into an uncertainty estimate that helps determine when the system should refrain from answering because its underlying knowledge appears unreliable [48]. Although the method primarily evaluates uncertainty in a single underlying black-box LLM, it illustrates how multi-agent interaction can support decisions about whether the system should commit to an answer at all.

Across these approaches, limitation awareness is generally represented through narrower signals such as missing information, disagreement, unstable reasoning, insufficient confidence, or the need for verification. These signals provide useful triggers for adapting or interrupting autonomous execution, but broader competence boundaries across tasks and operating conditions remain comparatively underexplored.

F.10 Extended Discussion of Equity in LLM-Based Multi-Agent Systems

F.10.1 Identity-Based and Interaction-Level Equity. Counterfactual evaluation provides one way to identify whether socially irrelevant attributes affect how otherwise comparable behavior is treated. Mayilvaghanan et al. [95] show that assessments can change when demographic, historical, or behavioral attributes are altered while substantive task information is held fixed. Although this setting evaluates individual LLMs rather than interacting agents, it establishes a useful baseline for detecting differential treatment. MALIBU extends this concern to multi-agent judging systems, showing that comparable responses can receive different evaluations when demographic identities are attached to them [99]. These findings indicate that identity information can shape not only outcomes for external users, but also how contributions are evaluated within a multi-agent process.

Differences in treatment become particularly consequential when they affect who is able to influence the collective decision. Li et al. [79] show that demographic personas can alter how readily an agent's position is accepted by others, how strongly agents maintain their own views, and how conformity develops across repeated interaction, even when the initial response content is controlled. This introduces an interaction-level form of inequity in which comparable contributions may receive different weight because of the identity associated with their source. Equity in LLM-MAS therefore concerns not only differential treatment in final decisions, but also whether participants have comparable opportunities to shape collective reasoning when differences in influence are not justified by the relevance or quality of their contributions.

F.10.2 Emergent and System-Level Equity. System-level equity cannot generally be inferred from the behavior of individual agents. Madigan et al. [92] show that collaboration can either reduce or amplify group disparities and that some multi-agent configurations produce substantially greater disparities than those observed for their constituent agents. Communication itself can alter the bias profile of the collective. Nguyen et al. [104] find that stereotypical responses can emerge during interaction, spread across agents, and become increasingly prevalent over subsequent rounds. Similarly, Okawa [106] show that conformity can transform relatively weak individual biases into strongly biased collective consensus, while agent heterogeneity and interaction conditions affect whether such consensus develops. Together, these results show that inequity can arise from the interaction process rather than merely being inherited from the underlying models.

System-level equity also extends beyond demographic disparities to the distribution of shared resources and collective gains. In GOVSIM, Piatti et al. [113] evaluate societies of LLM agents managing common resources and measure

inequality using a Gini-based metric. Their experiments show that interaction conditions that undermine cooperation can also worsen resource distribution, with the introduction of a greedy participant degrading both sustainability and equality. Similar distributional questions arise in LLM stakeholder negotiation, where inequality in negotiated gains can be measured separately from whether agents succeed in reaching an acceptable agreement [49]. These settings reinforce the distinction between collective success and equitable collective outcomes.

The structure through which agents interact can further determine whether disparities are reinforced or attenuated. Agent ordering, communication connectivity, and aggregation procedures affect which contributions receive greater influence and can therefore alter fairness-related outcomes even when the participating models remain unchanged [9]. Broader work on fairness in agentic AI similarly argues that fairness should be treated as a dynamic system property shaped by agent interactions, incentives, and correction mechanisms rather than as a constraint applied only to an isolated decision [121]. An apparently equitable outcome under one coordination structure may therefore fail to persist under another, making evaluation across multiple interaction configurations important.

F.10.3 Evaluating and Mitigating Inequity. Equity interventions operate at different stages of the decision process. Collaborative fairness auditing uses multiple auditing agents to coordinate queries to a black-box model and improves estimation of demographic-parity gaps under limited query budgets [36]. This illustrates how multi-agent coordination can be used not only as an object of fairness analysis but also as a mechanism for improving fairness measurement. REQUAL-LM addresses a different stage of the pipeline by applying bias-aware weighting over repeated LLM samples to select an output that remains representative of the model's response distribution while reducing demographic disparities [43]. Although REQUAL-LM does not model interacting agents, it shows how the aggregation mechanism itself can incorporate equity rather than treating fairness as independent of how a final response is selected.

In LLM-MAS, mitigation can also be incorporated directly into the workflow. MOMA uses specialized agents to modify social-group information before task execution and seeks to reduce social bias while limiting the associated loss in task performance [168]. Other agentic fairness frameworks introduce fairness constraints, bias-sensitive incentives, and adaptive correction mechanisms within the coordination process [121]. Taken together, these approaches show that inequity can be addressed through several intervention points, including fairness auditing, input transformation, aggregation, incentive design, and changes to the interaction process itself. The appropriate intervention depends on where the disparity originates, whether in the underlying model, the treatment of participants during interaction, the aggregation of their contributions, or the distribution of the resulting benefits and burdens.

F.11 Extended Discussion of Ethical Norm Adherence

F.11.1 Norm Compliance and Constraint Preservation. MAC-Bench directly evaluates whether LLM-based multi-agent systems satisfy legal, security, privacy, ethical, and organizational requirements while completing assigned tasks [192]. Its experiments demonstrate that task success and compliance can diverge substantially and that the coordination architecture affects this relationship. For GPT-4o under high operational pressure, a hierarchical AutoGen configuration achieves 96.8% task success but only 38.5% compliance, while the ChatDev configuration achieves 89.1% task success and 70.3% compliance. The results therefore illustrate that high collective task performance can coexist with substantial violations of the requirements governing execution.

A related problem arises when constraints are initially present but lose their operational effect as execution progresses. Li et al. [83] describe this as *constraint drift*, in which safety critical requirements can be lost, distorted, or weakened through memory, authority delegation, information flow, auditing, or optimization. Their replay analysis shows that

inspection of the final output can conceal substantial internal violations. An output filter reduces visible information leakage to 2.9%, while internal exposure remains 68.8%. Their Constraint State Governance framework addresses this problem by representing constraints as explicit execution state and checking their continued validity before consequential actions.

EnviSmart provides a related architectural example [53]. Governance rules, authorization constraints, and safety boundaries are externalized as persistent behavioral artifacts, while agent handoffs are treated as explicit trust boundaries. Actions with irreversible consequences proceed only after deterministic validation and approval. Although the framework is primarily motivated by operational reliability, it illustrates how normative requirements can be preserved across handoffs without requiring each downstream agent to reconstruct them from natural language context.

F.11.2 Moral, Professional, and Interactional Norms. Norm adherence may also be evaluated against socially grounded moral expectations rather than explicit obligations or prohibitions. FairMindSim examines ethical dilemmas involving unfair resource allocation and models the relationships among beliefs, rewards, emotions, and behavior [75]. Its experiments compare human and LLM responses and report that GPT-4o exhibits relatively stable beliefs oriented toward fairness and less reward driven behavior under the studied conditions. Unlike formal compliance benchmarks, the framework evaluates whether behavior remains consistent with expectations concerning fairness and justice.

Professional standards provide another source of normative criteria. MedSentry evaluates medical LLM-MAS using an Ethics Evaluator grounded in the AMA Principles of Medical Ethics and considers risks including harmful medical advice, privacy violations, discriminatory care, and harms to vulnerable populations [23]. Its experiments further show that safety outcomes depend on interaction topology and that malicious agents can propagate unsafe behavior through the collective process. The study therefore illustrates how standards specific to a professional domain can provide external criteria for evaluating multi-agent conduct.

Norms can also concern the manner in which agents interact. Interactional Fairness operationalizes interpersonal expectations through respectful and dignified communication and informational expectations through the clarity and adequacy of explanations [11]. In resource allocation experiments, respectful communication and clear justification affect whether proposals are accepted even when the underlying allocation remains unchanged. This provides a behavioral interpretation of norm adherence without assuming that LLM agents themselves possess moral experiences or intentions.

Normative principles may additionally be incorporated directly into collective reasoning. de Cerqueira et al. [35] use specialized development and ethics roles to introduce fairness evaluation, regulatory considerations, and other ethical requirements into AI software generation. In a different setting, Piatti et al. [113] show that prompting agents to apply a universalization principle can improve sustainable cooperation in common pool resource environments. These approaches illustrate how normative principles can influence the interaction process itself rather than serving only as criteria for retrospective evaluation.

F.11.3 Verifying Ethical Norm Adherence. Because violations can occur during execution without appearing in the final output, trajectory evidence is important for determining whether applicable constraints were actually respected. HarnessAudit evaluates agent systems against predefined tool, resource, permission, and information flow boundaries throughout execution [89]. Relative to a controlled single-agent setting, multi-agent execution reduces tool adherence from 0.91 to 0.64 and resource adherence from 0.85 to 0.63, while information sharing adherence reaches only 0.58. These results show that collaboration can enlarge the surface over which violations occur even when the overall task is completed successfully.

Although HarnessAudit primarily contributes to Auditability, its results are also relevant to ethical norm adherence because they demonstrate why final output inspection is insufficient for verifying compliance. Together with constraint preservation approaches, this work suggests that responsible LLM-MAS require both mechanisms that maintain applicable norms during execution and evidence that allows their continued observance to be checked.