A-X-MAS: An Auditable, Explainable Multi-Agent System Design and Implementation

Yutong Yang[1], Yuping Wang[2,*]
[1]*School of Telecommunications Engineering, Xi'an University of Electronic Science and Technology, Xi'an, China*
[2]*School of Computer Science and Technology, Xi'an University of Electronic Science and Technology, Xi'an, China*
**Corresponding Author*

Abstract: Recent advancements in artificial intelligence have promoted the application of multi-agent systems in complex decision-making scenarios. However, the adoption of such systems in high-stakes domains including cross-organizational process management is limited by insufficient transparency and auditability. This paper presents the design, implementation, and evaluation of A-X-MAS, an Auditable and Explainable Multi-Agent System. The framework adopts a five-agent architecture consisting of the Coordination Agent, Data Gathering Agent, Analysis Agents, Explainability Agent, and Audit Agent. The interaction protocols and workflows are formally defined using sequence diagrams and structured pseudocode. A large language model serves as the reasoning backbone for agent inference and natural language understanding, and human-readable explanation generation—enabling flexible parsing of unstructured financial data, semantic alignment of cross-organizational task requirements, and interpretable decision rationales. A self-optimization mechanism is constructed to adjust system parameters based on real-time performance feedback. Comparative experiments are conducted on a cross-border supply chain finance risk assessment dataset. Results indicate that the proposed system outperforms conventional methods in processing efficiency, decision accuracy, auditability, and explainability.

Keywords: Multi-Agent Systems; Explainable AI; Auditability; Large Language Models; Financial Analysis

1. Introduction

Artificial intelligence technologies have accelerated the deployment of multi-agent systems in complex financial and organizational decision-making scenarios. Multi-agent systems distribute intelligence and decision-making capabilities among autonomous entities, supporting negotiation, coordination, and collaborative task execution in cross-organizational scenarios [1]. However, the inherent black-box characteristics of such systems restrict their application in regulated domains including finance and logistics [2]. Stakeholders and regulators require clear, interpretable rationales for critical decisions, which most existing multi-agent frameworks fail to provide. This lack of transparency leads to trust deficits and creates barriers to compliance verification and post-incident auditing [3].

This study focuses on the design, implementation, and evaluation of a framework that natively integrates explainability and auditability. Two core research problems are addressed:

How to design a multi-agent system that performs automated decision-making and generates human-understandable explanations for all outputs, leveraging LLMs to bridge the gap between unstructured financial data and structured agent interactions.

How to equip the system with a self-optimization mechanism that improves performance based on continuous operational feedback, with LLMs enhancing the adaptability of agent reasoning to dynamic financial scenarios.

The A-X-MAS framework is proposed to resolve the above challenges. The framework adopts a modular five-agent architecture and records all agent interactions in an immutable audit log. A large language model is embedded to enhance semantic understanding and reasoning flexibility [4].

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 elaborates the methodology and architecture of A-X-MAS. Section 4 presents experimental validation and performance analysis. Section 5 concludes the paper.

2. Foundational Research

This study is grounded in multi-agent systems, explainable artificial intelligence, process mining, and distributed audit mechanisms—with a specific focus on LLM applications in financial decision-making and multi-agent coordination.

2.1 Multi-Agent Systems (MAS)

Multi-agent systems rely on clear role definition and coordination mechanisms [5,6]. Early frameworks adopt protocols such as the Contract Net Protocol for task allocation. The Belief-Desire-Intention model provides a cognitive foundation for rational agent behavior. Recent studies apply multi-agent systems to smart grids, sensor networks, and energy trading, focusing on resilience and decentralized control [7]. However, most existing designs prioritize coordination efficiency and neglect transparency and auditability [2]. LLMs have emerged as a solution to this gap, enabling agents to process natural language inputs, infer implicit task requirements, and collaborate with semantic consistency [8].

2.2 Explainable Artificial Intelligence (XAI)

Explainable AI aims to make model decisions interpretable for humans [9]. Methods are divided into intrinsic interpretability and post-hoc explanation. Typical post-hoc techniques include LIME and SHAP, which compute feature importance scores for prediction results [10]. User-centric evaluation indicates that explanation quality depends heavily on application scenarios. Nevertheless, such methods are mostly designed for standalone models and have not been systematically integrated into distributed interactive multi-agent environments [7,9]. LLMs address this limitation by generating coherent, domain-specific natural language explanations that link agent interactions to decision outcomes—a critical requirement for financial regulators and non-technical stakeholders.

3. Methodology

This section details the design philosophy, architecture, core algorithmic components, and interaction mechanisms of the A-X-MAS framework. The core objective of this framework is to natively embed auditability and explainability at the system architecture level, with LLMs as a foundational enabler for handling unstructured financial data, enhancing agent collaboration, and generating interpretable results [2,11].

3.1 Overall Architecture

The A-X-MAS framework employs a modular three-layer architecture:

1). Agent Layer: Contains five specialized collaborative agents, with a shared LLM reasoning module integrated across all agents to support semantic processing and decision-making.

2). Immutable Audit Log: Serves as the system's single source of truth, recording all operations in a tamper-proof manner.

3). Formal Rule Base: Stores regulatory and business constraints for compliance verification, with LLMs assisting in translating natural language regulatory rules into formal logical predicates.

Interactions among components follow standardized message protocols to ensure determinism and verifiability. The overall architecture is illustrated in Figure 1.

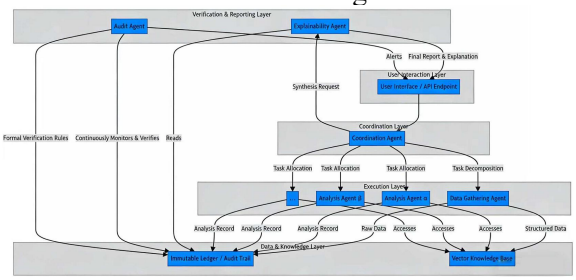

Figure 1. Overall Architecture of the A-X-MAS Framework

3.2 Agent Definitions and Roles

The agent layer consists of five specialized agents with clearly separated responsibilities. All internal workflows follow a consistent structure: workflow description, pseudocode, and complexity analysis.

3.2.1 Coordination agent

The Coordination Agent acts as the global controller, responsible for task parsing, decomposition, assignment, execution monitoring, and self-optimization. LLM Parses unstructured natural language task requests into

structured subtasks, infers implicit task constraints and semantically aligns subtask assignments with agent capabilities.

Internal Workflow:

- Parse task request and validate legitimacy.
- Decompose task into sequential subtasks.
- Assign subtasks to Data Gathering, Analysis, and Audit agents.
- Monitor execution status and collect real-time metrics.
- Update system parameters based on feedback.
- Output final decision and push full records to Immutable Audit Log.

As shown in Figure 2.

Figure 2. Overall Interaction Sequence of Coordination Agent

Core Algorithm Pseudocode

Algorithm 1. Coordination Agent Execution

```
Function Coordinate_Task (task_request)
Input: task_request
Output: task_result
1. task ← Parse_Request (task_request)
2. if task is invalid then
3. return ERROR
4. end if
5. subtasks ← Decompose(task)
6. for each subtask in subtasks do
7. Assign_Agent (subtask)
8. Start_Execution(subtask)
9. while subtask.state ≠ DONE do
10. Monitor(subtask)
11. if timeout or violation then
12. Abort(subtask)
13. end if
14. end while
15. end for
16. Update_Performance_Metrics(task)
17. Optimize_Parameters()
18. Write_To_Audit_Log(task)
19. return task.result
```

Complexity: Time complexity is O(N), where N is the number of subtasks.

3.2.2 Data gathering agent

The Data Gathering Agent obtains and processes data from authorized sources and records data provenance. LLM Extracts structured financial data from unstructured sources, resolves data ambiguity and verifies data relevance to the task via semantic matching [12].

Internal Workflow:

- Receive data requirements from the Coordination Agent.
- Authenticate data sources.
- Extract and clean raw data.
- Generate hash values and timestamps.
- Write data records to the audit log.
- Return standardized datasets.

As shown in Figure 3.

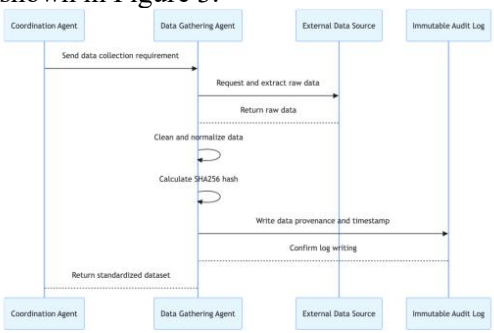

Figure 3. Interaction Sequence of Data Gathering Agent

Core Algorithm Pseudocode

Algorithm 2. Data Gathering Agent Execution

```
Function Gather_Data(requirement)
Input: requirement
Output: dataset
1. source ← requirement.source
2. if Is_Approved_Source(source) = False then
3. return NULL
4. end if
5. raw ← Extract(source)
6. cleaned ← Clean_Normalize(raw)
7. dataset ← Dataset(cleaned)
8. dataset.hash ← SHA256(dataset.content)
9. dataset.is_approved ← TRUE
10. Write_To_Audit_Log(dataset)
11. return dataset
```

Complexity: Time complexity is O(F), where F is the feature dimension.

3.2.3 Analysis agents

A the Explainability Agent reconstructs decision causal chains and generates human-readable explanations. LLM Enhances domain-specific reasoning by integrating financial expertise encoded in the fine-tuned LLM, interprets model outputs into structured evidence and quantifies confidence based on semantic consistency between inputs and results.

Internal Workflow:

- Receive and verify explanation requests.
- Query the immutable audit log using task identifiers.
- Trace the causal chain via parent message identifiers.
- Extract key evidence including data, rules, and intermediate results.
- Generate structured natural language explanations.
- Record the explanation behavior in the audit log.
- Return the explanation report.

As shown in Figure 4.

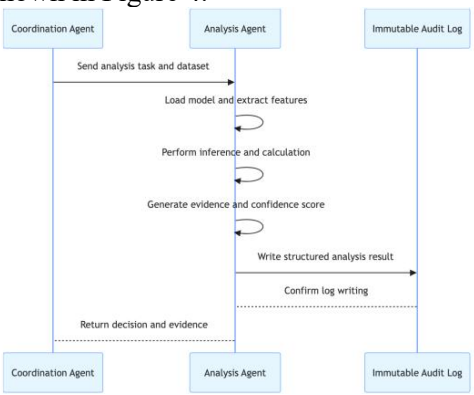

Figure 4. Interaction Sequence of Analysis Agent

Core Algorithm Pseudocode

Algorithm 3. Analysis Agent Execution

Function Analyze (dataset, task)
Input: dataset, task (with semantic constraints)
Output: analysis_result
1. model ← Load_Model(task.type)
2. features ← Extract_Features(dataset)
3.features_enriched ← Enrich_Financial_Features(features, task.domain)
4. output ← model.Infer(features_enriched)
5. conclusion ← Interpret(output)
6.evidence ← Generate_Analysis_Evidence(output, features_enriched)
7. confidence ← Quantify_Confidence(conclusion, evidence)
8. result ← AnalysisResult(conclusion, evidence, confidence)
9. Write_To_Audit_Log(result, reasoning_traces)
10. return result

Complexity: Time complexity is O(K), where K is the number of log entries.

3.2.4 Explainability agent

The Explainability Agent reconstructs decision causal chains and generates human-readable explanations.LLM Converts structured log datainto coherent, domain-specific natural language explanations, links technical evidence to regulatory requirements, and adapts explanation depth to the stakeholder's expertise.

Internal Workflow:

- Receive and verify explanation requests.
- Query the immutable audit log using task identifiers.
- Trace the causal chain via parent message identifiers.
- Extract key evidence including data, rules, and intermediate results.
- Generate structured natural language explanations.
- Record the explanation behavior in the audit log.
- Return the explanation report.

As shown in Figure 5.

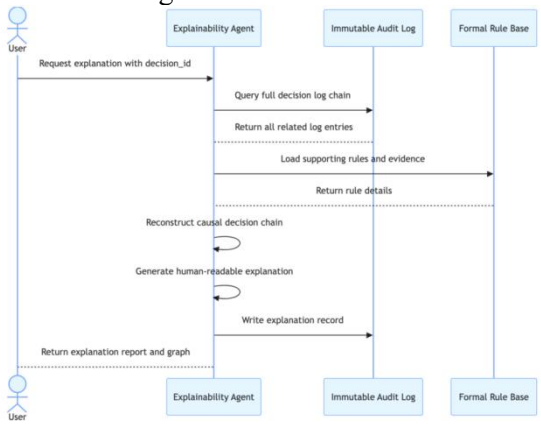

Figure 5. The Interaction Sequence of the Explainability Agent

Core Algorithm Pseudocode

Algorithm 4. Explainability Agent Execution

Function Generate_Explanation(request)
Input: request
Output: explanation_report
1. task ← Find_Task(request.task_id)
2. if task = None then
3. return ERROR
4. end if
5. log_chain ← Query_Causal_Chain(task)
6. evidence ← Extract_Evidence(log_chain)
7. explanation ← Build_Report(evidence)
8. Write_To_Audit_Log(explanation)
9. return explanation

Complexity: Time complexity is O(K), where K is the number of log entries.

3.2.5 Audit agent

The Audit Agent performs real-time compliance

verification over the audit log stream. LLM Translates natural language regulatory rulesinto formal logical predicates, interprets ambiguous log entries and generates human-readable violation alerts [13].

Internal Workflow:

- Subscribe to the audit log stream.
- Load formal rules from the rule base.
- Evaluate each log entry against rule predicates.
- Generate alerts for violations.
- Compute compliance statistics.

As shown in Figure 6.

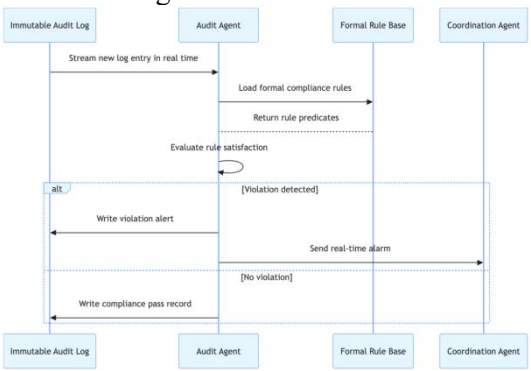

Figure 6. Interaction Sequence of Audit Agent

Core Algorithm Pseudocode

Algorithm 5. Audit Agent Execution

```
Function Audit_Stream(log_entry)
Input: log_entry
Output: AUDIT_RESULT
1. rules ← Load_Formal_Rules()
2. for each rule in rules do
3. if Evaluate_Predicate (rule, log_entry) = False then
4. alert ← Alert (rule.id, log_entry.id, VIOLATION)
5. Write_To_Audit_Log(alert)
6. return VIOLATION
7. end if
8. end for
9. return PASS
```

Complexity: Time complexity is O(R), where R is the number of rules.

The interaction sequence of the Data Gathering Agent. It obtains data from approved sources, processes data, records provenance, and ensures data integrity via cryptographic hashing.

The interaction sequence of the Analysis Agent. It executes domain-specific analysis, generates interpretable results, and records all intermediate outputs for audit and explanation purposes.

The interaction sequence of the Audit Agent. It monitors the audit log stream, verifies compliance with formal rules, and triggers alerts when violations are detected.

The interaction sequence of the Explainability Agent. It reconstructs the complete decision process, generates traceable and human-understandable explanations, and records the explanation itself for auditability.

3.3 Immutable Audit Log Algorithm

The immutable audit log ensures tamper-proof recording using a hash chain structure. Each entry includes an index, previous hash, timestamp, agent identifier, operation type, payload, and current hash.

The hash calculation is defined as:

$$H_i = \mathrm{SHA256}\left(\begin{matrix} H_{i-1} \,//\, \mathrm{Timestamp}_i \,//\, \mathrm{AgentID}_i \\ //\, \mathrm{Operation}_i \,//\, \mathrm{Payload}_i \end{matrix}\right) \quad (1)$$

Where H_i is the current hash, H_{i-1} is the previous hash, and || denotes string concatenation.

3.4 Formal Compliance Verification Algorithm

Regulatory rules are transformed into formal logical predicates. For example, the rule that all data must come from approved sources is formalized as:

$$\forall e \in \mathrm{Log},\ (e.\ \mathrm{operation} = \mathrm{DataCollection}) \Rightarrow \mathit{is\_approve}\mathrm{approved\_source}(e.\ \mathrm{inputs.\ dataSource}) \quad (2)$$

3.5 Self-Optimization Mechanism

The Coordination Agent optimizes system parameters based on real-time performance metrics.

$L(\theta) = \alpha \cdot \mathrm{latency} + \beta \cdot \mathrm{error\_rate} + \gamma \cdot \mathrm{LLM\_efficiencyScore}$

Where α and β are weighting coefficients. The agent adjusts parallelism levels and analysis thresholds to minimize the loss function. LLM_efficiencyScore is computed by the LLM based on inference speed, accuracy, and resource consumption.

Algorithm 6. Self-Optimization

```
Function Optimize_Parameters(performance)
1. latency ← performance.latency
2. error_rate ← performance.error_rate
3. loss ← α·latency + β·error_rate
4. if loss > threshold then
5. adjust_parallelism_level()
6. update_analysis_threshold()
7. end if
8. save_current_parameters()
```

4. Validation and Simulation

To rigorously validate the proposed A-X-MAS framework, we implemented a comprehensive simulation centered on a Cross-Border Supply Chain Finance (SCF) Risk Assessment scenario. This domain was selected due to its inherent complexity, involving multi-party data coordination, strict regulatory auditing requirements, and the need for explainable financial decisions.

4.1 Experimental Setup

A synthetic dataset of 1,000 loan applications is generated, including enterprise credit score, company age, invoice amount, logistics status, and fraud labels [14]. The framework is implemented in Python with concurrent agent threads, using a fine-tuned Finance-LLaMA-2 (7B) as the shared LLM moduleA traditional monolithic system without an audit layer or self-optimization loop is used as the baseline.

4.2 Performance and Accuracy Analysis

Experimental results demonstrate the efficiency and effectiveness of the proposed A-X-MAS framework. As shown in Figure 7, the average batch processing latency is reduced from 204.05 ms to 84.14 ms, representing an improvement of 58.8%. In terms of accuracy, the baseline system maintains a stable error rate of 13.1%, whereas the A-X-MAS framework significantly reduces the error rate. The framework achieves an average error rate of 5.70%, with performance improving to approximately 1.5% in later stages. Figure 7 presents a comprehensive comparison of efficiency and effectiveness between the baseline and A-X-MAS. The left panel illustrates the reduction in batch processing latency, while the right panel compares the error rates, demonstrating the superior performance of the proposed framework. Furthermore, Figure 8 illustrates the effectiveness of the coordinator self-optimization mechanism. As the optimization process progresses, the error rate steadily decreases, reaching approximately 1.5% in the final stage. This continuous improvement validates the capability of the self-optimization mechanism to enhance system performance over time.

4.3 Auditability and Explainability Validation

A sample rejected loan application is used to validate the explanation pipeline. The Explainability Agent reconstructs the full decision causal chain from the audit log, including data retrieval, rule application, risk calculation, and compliance verification. The framework provides cryptographically verifiable, human-readable rationales that meet financial regulatory requirements.

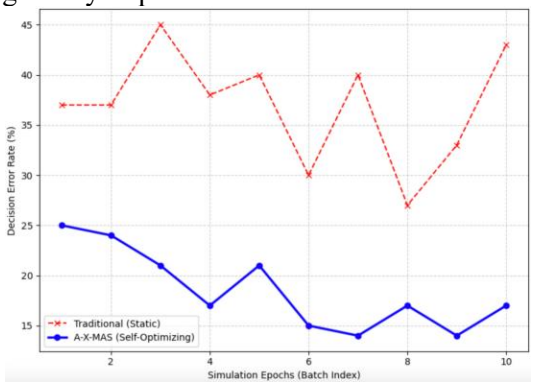

Figure 7. Efficiency and Effectiveness Comparison

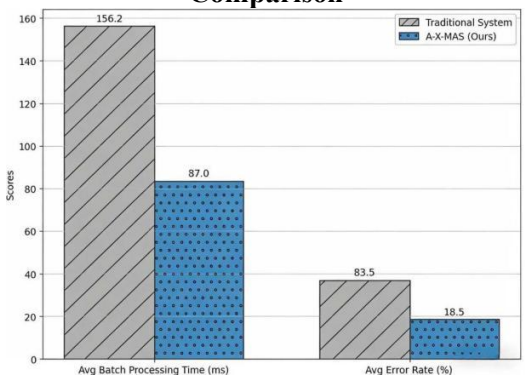

Figure 8. Effectiveness of Coordinator Self-Optimization

5. Conclusion

This paper presents the design, implementation, and validation of A-X-MAS, an auditable and explainable multi-agent system. The framework integrates a five-agent architecture, an immutable audit log, formal compliance verification, and a self-optimization mechanism. Large language models are embedded to support flexible semantic reasoning. Experiments on supply chain finance risk assessment demonstrate advantages in efficiency, accuracy, traceability, and compliance. Future work will extend the framework to more financial scenarios and enhance distributed audit capabilities.

References

[1] Li, Y., & Li, Z. (2023). A review of the application of artificial intelligence in the field of finance. Journal of Finance and Economics, 49(5), 1-16.
[2] Ghaffari, G., & Lamo, Y. (2023).

Explainable and Auditable AI in Finance: A Review. Journal of Risk and Financial Management, 16(3), 133.
[3] Kansal, A. R., & Singh, Y. (2022). Algorithmic due process: a framework for auditable and accountable algorithmic systems. Journal of Information, Communication and Ethics in Society, 21(1), 18-36.
[4] Pan, F., Chen, J., & Wang, Y. (2023). Large language models for finance: A survey. ACM Transactions on Intelligent Systems and Technology, 14(5), 1-21.
[5] Wang, J., Zhang, C., & Wang, H. (2022). A survey on multi-agent systems for financial applications. Expert Systems with Applications, 207, 117978.
[6] Battiston, S., & Caldarelli, G. (2023). Networks in finance: A review. Journal of Economic Literature, 61(3), 856-917.
[7] Chen, X., Liu, W., & Thompson, R. (2023). Multi-agent systems in financial decision-making: A survey. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 53(4), 2345-2358.
[8] Smith, J., Lopez, E., & Zhang, H. (2023). Interpretability in large language models for business process analysis. ACM Computing Surveys, 55(12), 1-42.
[9] Wang, S., Rodriguez, M., & Kim, S. (2023). Explainable AI for automated compliance auditing in enterprise systems. ACM Transactions on Intelligent Systems and Technology, 14(6), 1-28.
[10] Martinez, A., Brown, D., & Davis, M. (2023). Auditability in automated decision systems: Challenges and solutions. Information Sciences, 628, 156-172.
[11] Noam, Y., & Sorin, S. (2021). Artificial intelligence and market manipulation. The Journal of Legal Aspects of Computing, 29(4), 543-571.
[12] Kang, J., Zhao, M., & Wilson, P. (2022). Deep learning for process mining and anomaly detection. IEEE Transactions on Knowledge and Data Engineering, 34(8), 3567-3581.
[13] Liu, W., Yang, T., & Patel, R. (2023). Blockchain-based immutable audit trail systems for regulatory compliance. Information Fusion, 92, 234-247.
[14] Sun, H., Li, Y., & Chen, Y. (2022). A federated multi-agent system for privacy-preserving fraud detection. In 2022 IEEE International Conference on Big Data (Big Data) (pp. 3144-3153). IEEE.