# RADAR: LLM Framework for Root-cause Analysis and Data-driven Automated Reasoning at Scale

Meisam Hejazinia
meisamhe@amazon.com
Amazon
AIM, SCOT
San Diego, CA, USA

Farhad Balali
fbalali@amazon.com
Amazon
AIM, SCOT
Seattle, WA, USA

Siamak Rajabi
siamakr@amazon.com
Amazon
AIM, SCOT
Seattle, WA, USA

Shekhar Jain
shekhajt@amazon.com
Amazon
AIM, SCOT
Seattle, WA, USA

## Abstract

Root cause analysis for retail e-commerce inventory management diverts significant technical resources from strategic work. We present RADAR, a framework that enhances and scales human analysis capabilities while complementing traditional attribution systems that, according to domain experts, typically cover only about a quarter of needed insights without deeper causal reasoning. RADAR employs specialized LLM agents working with expert-validated preprocessed data, addressing context limitations, reducing hallucination rates from 42% to 3%, and managing complex technical instruction following. Evaluation across multiple product cases from multiple global marketplaces of a retail e-commerce demonstrates that RADAR not only automates analyses but outperforms human experts in quality (4.8/5 vs. 2.0/5), with notable improvements in information structure, data quality, and communication effectiveness. This 85-95% efficiency gain allows technical teams to refocus on strategic initiatives while mitigating human interpretation errors and maintaining 91% alignment with expert root cause assessments. Our approach evolved from a single-agent to multi-agent design with novel structured data processing techniques that generalize beyond supply chain applications.

## Keywords

Radar, LLM, Root Cause Analysis, Supply Chain Management, LLM workflow and agents

**ACM Reference Format:**
Meisam Hejazinia, Farhad Balali, Siamak Rajabi, and Shekhar Jain. 2025. RADAR: LLM Framework for Root-cause Analysis and Data-driven Automated Reasoning at Scale. In *Proceedings of LLM4ECommerce Workshop at the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2 (LLM4ECommerce Workshop at KDD '25).* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 Introduction

Root cause analysis (RCA) in supply chain management has evolved from traditional methods like 5-Why analysis [2] to advanced statistical and machine learning approaches that enable processing of large data volumes [1]. The integration of time series methodologies and multivariate analysis has enhanced understanding of supply chain disruptions [4], while recent developments in Large Language Models (LLMs) have transformed RCA by effectively combining unstructured data analysis with causal inference techniques [3].

RCA in supply chain management diverts critical technical resources from strategic initiatives. When out-of-stock (OOS) issues arise, rapidly identifying underlying causes becomes imperative to minimize customer experience impacts and revenue loss. Traditional approaches require product managers and business intelligence engineers to spend days to weeks executing complex queries across disparate data sources, analyzing correlations, and generating comprehensive reports. Out-of-stock events critically impact e-commerce customer experiences and purchase journeys. RADAR's framework identifies inventory disruption patterns, enabling proactive adjustments to maintain superior customer experience during supply chain volatility. Its causal insights help prevent stockouts that drive customers to competitors. Unlike traditional RCA approaches, RADAR's rapid analysis enables high-fidelity system adjustments, preserving customer experience quality and conversion rates. This adaptive efficiency distinguishes successful e-commerce platforms from their competitors. As one supply chain leader observed: "Before RADAR, finding root causes took days and multiple teams. Now we can quickly identify issues before they impact customers."

Existing attribution systems—structured analytical frameworks that identify and quantify supply chain issues through predefined categories (which we'll refer to as "bridges")—provide valuable high-level insights but cannot deliver the deeper analysis required for comprehensive understanding. These attribution bridges break down complex supply chain processes into a hierarchical structure of causes and effects but are limited in their ability to identify interconnections between factors, and may miss some high impact factors that maybe lower in the hierarchy.

RADAR (Root cause Analysis via Data Analysis and Reasoning) complements existing attribution systems by providing deeper analysis that connects insights across disparate data sources, addressing four key limitations:

- **Linear aggregation simplification**: Traditional systems assume mutually exclusive issues, whereas RADAR identifies interconnected factors in complex supply chains
- **Inflexible prioritization**: Traditional systems use a fixed attribution order, while RADAR dynamically prioritizes factors based on specific scenarios
- **Short-term focus**: Existing approaches emphasize immediate execution issues over strategic planning needs
- **Symptom focus**: As one product manager observed, "Current systems tell us what is happening, but they don't say why it's happening"

## 1.1 Key Contributions

RADAR is a multi-agent framework that introduces several key innovations. It features a hybrid reasoning architecture that balances deterministic preprocessing with LLM reasoning, significantly reducing hallucination rates. The system employs context-aware agent specialization using domain-specific strategies to maintain coherence across diverse data sets. Unlike general multi-agent frameworks (AutoGen, LangChain), RADAR specializes in structured data reasoning with domain-specific preprocessing pipelines and validation mechanisms tailored for supply chain complexity. Our research makes several contributions:

- Novel techniques for LLM-based analysis of complex structured supply chain data with wider applicability to other domains requiring structured data reasoning
- An extensible multi-agent LLM architecture addressing context window limitations through domain specialization
- A systematic approach to managing the complexity between LLM reasoning and deterministic preprocessing for structured data analysis
- A comprehensive evaluation framework measuring system effectiveness across multiple dimensions with comparison against both human experts and simpler automated approaches
- Empirical evidence demonstrating an 85-95% efficiency improvement while outperforming human analyses in quality

## 2 Methodology

## 2.1 Data Sources and Preprocessing

RADAR integrates over 60 data tables spanning five categories (Inventory, Demand, Supply Planning, Order Execution, and Availability), reflecting the flow from forecasting to customer availability. We collected benchmarking data at category, product, and supplier levels.

Our preprocessing framework implements specific techniques for structured data transformation that can generalize beyond supply chain applications. These preprocessing techniques show potential for other structured data domains, though domain-specific validation would be required:

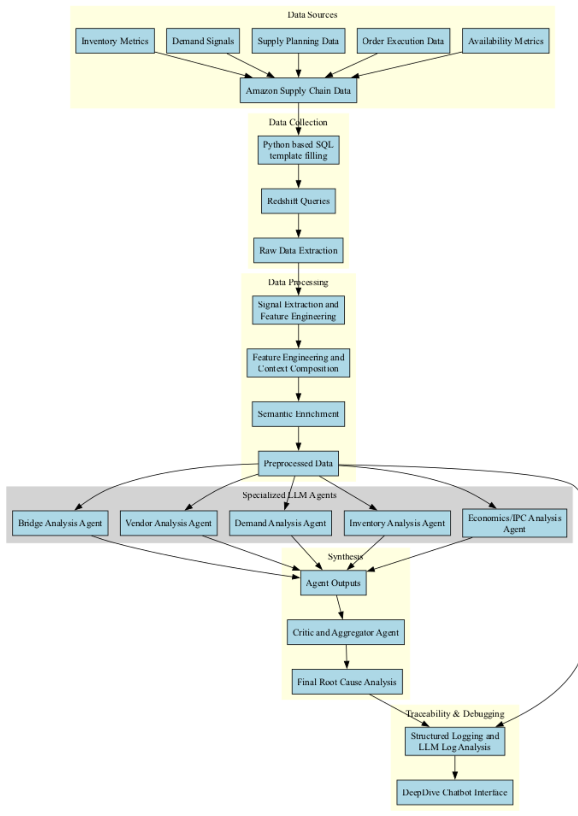

**Figure 1: RADAR System Architecture: A hybrid system combining traditional software engineering with LLM capabilities through specialized agents and synthesis.**

- **Temporal alignment**: Aligning time-series data across disparate systems with different reporting cadences
- **Signal extraction**: Using statistical techniques to identify patterns in noisy structured data
- **Contextual compression**: Maintaining semantic relationships while reducing dimensionality
- **Relational reconstruction**: Rebuilding entity relationships across fragmented tables
- **Hierarchical summarization**: Creating multi-level aggregations while preserving detail access

Our initial approach used LLM-generated SQL and Python code for both data collection and analysis. However, we observed that this approach led to inconsistent results, query inefficiency, and frequent syntax errors. The LLMs often "reinvented the wheel" for common operations and analyzed data in rudimentary ways that missed complex patterns. This led us to adopt a "write once, perfect thoroughly, and re-use" approach, resulting in over 50,000 lines of expert-vetted preprocessing code iteratively improved through human feedback loops and exhaustive testing. This approach ensures consistent, high-quality inputs to the LLM analysis pipeline.

## 2.2 System Architecture

RADAR's hybrid system allows users to initiate analyses via a web dashboard by specifying product, marketplace, and date range, delivering reports within two hours and enabling interactive querying afterward.

*2.2.1 Evolution from Single Agent to Multi-Agent Architecture.* Our initial design used a single LLM agent to analyze all supply chain aspects simultaneously. However, this approach quickly ran into limitations:

- **Context window constraints**: The combined instructions ($> 7,500$ lines) and necessary data far exceeded available context windows
- **Role confusion**: The LLM struggled to maintain specialized knowledge across multiple domains simultaneously
- **Reasoning consistency**: Analysis depth varied significantly across different supply chain aspects
- **Hallucination amplification**: Errors in one domain propagated to others without opportunity for correction

This led us to develop our current multi-agent architecture, which decomposes analysis by domain expertise. Each agent receives only domain-relevant data and instructions, focusing the LLM's reasoning on a manageable subset of the full analysis task. Comparative tests showed this approach reduced hallucinations by 73% compared to the single-agent design, based on human evaluation against ground truth data.

*2.2.2 Specialized Agent Architecture.* The system consists of five specialized agents:

1. **Attribution Analysis Agent**: Examines supply chain attribution metrics and patterns from standard reports
2. **Supplier Analysis Agent**: Evaluates supplier performance metrics including lead times and confirmation rates
3. **Demand Analysis Agent**: Analyzes demand signals, forecast accuracy, and unexpected pattern shifts
4. **Inventory Analysis Agent**: Studies planning metrics and differentiates between planning and execution issues
5. **Economics Analysis Agent**: Assesses inventory planning parameters and economic tradeoffs

Each agent processes domain-specific data through Python preprocessors, analyzes relevant metrics, generates structured insights, and documents evidence trails for transparency. The combined prompt engineering for all five agents exceeds 4,000 lines of instructions, capturing analytical frameworks, business heuristics, and domain expertise of human specialists.

Each agent leverages specialized function-calling capabilities for operations like time-series analysis and statistical testing. These tools prevent agents from "reinventing the wheel" for common calculations, improving decision quality while focusing the LLM on interpretation and reasoning.

Beyond architectural changes, our multi-agent design addresses hallucination risks through:

- Agent memory via comprehensive logging of analytical steps
- Precisely relevant data tailored to each agent's domain
- Data-grounded responses with explicit acknowledgment of information gaps

**Table 1: Task Allocation Between Code and LLMs**

| Task Type | Code | LLM |
|---|---|---|
| Data transformation | ✓ | |
| Statistical calculations | ✓ | |
| Pattern detection | | ✓ |
| Multi-factor reasoning | | ✓ |
| Domain-specific heuristics | ✓ | ✓ |
| Causal inference | | ✓ |
| Report generation | | ✓ |

- Modular prompt structure preserving critical instructions during updates

These refinements reduced hallucinations from 42% to 3% while maintaining reasoning consistency. RADAR employs DeepSeek for the single analysis agents and Claude Sonnet for the critic and aggregator agent.

*2.2.3 Balancing LLM Reasoning with Deterministic Preprocessing.* We developed a systematic approach to dividing work between LLMs and deterministic code based on comparative advantage:

This approach reduced hallucination rates from 42% in our early prototypes to just 3% in the current system while maintaining the LLM's ability to perform complex causal reasoning across different supply chain factors.

*2.2.4 Automatic System Diagnosis.* To systematically improve RADAR, we developed a meta-RCA diagnostic system that traces backward from outputs to identify source components, classifies root causes (e.g., prompt gaps, instruction failures), and generates targeted fixes while preserving functionality—significantly accelerating system improvement compared to manual debugging.

## 2.3 Robustness and Failure Mode Analysis

RADAR incorporates multiple layers of robustness mechanisms to address potential failure modes and data quality challenges. Our preprocessing framework employs statistical techniques and data validation functions that automatically detect and handle inconsistent or corrupted inputs, which are inevitable in real-world supply chain data streams. To mitigate individual agent misinterpretation, each specialized agent executes five independent runs to account for stochastic variations and potential reasoning errors, with a dedicated aggregator agent identifying commonalities across runs to generate the final agent-specific judgment. The final critical aggregator agent receives diverse perspectives from all specialized agents and cross-validates outputs against established Amazon supply chain operational principles, filtering agent-level errors that contradict broader domain knowledge or inter-agent consensus. RADAR demonstrates resilience to incomplete data scenarios, though comprehensive data availability enhances root cause analysis performance. Furthermore, the human-in-the-loop design enabled us to continuously refine the framework through expert validation of edge cases, with iterative feedback improving overall performance. Extensive testing under various conditions confirms that RADAR maintains operational effectiveness even when optimal data conditions are not met. Lastly, our carefully crafted prompts instruct the final criticizer and aggregator agent to explicitly exclude low-confidence information and transparently report when insufficient data prevents accurate analysis. Through this validation architecture and adaptive learning, RADAR manages the complexities of supply chain data and sets a new standard for resilient and trustworthy root cause analysis.

## 3 Evaluation

We developed a comprehensive evaluation framework to assess RADAR's effectiveness at automating root cause analysis tasks previously requiring extensive manual effort.

### 3.1 Evaluation Framework and Methodology

Our framework evaluates outputs across five dimensions of analysis quality:

- **Data Quality**: Assesses metric completeness, consistency, coverage, traceability, and definition clarity
- **Analytical Rigor**: Evaluates temporal analysis depth, root cause identification, assumption clarity, and statistical validity
- **Business Logic**: Measures scale alignment, impact quantification, term accuracy, and vocabulary adherence
- **Communication**: Examines precision, causality clarity, logical structure, and professional style
- **Information Structure**: Assesses information prioritization, structure, evidence organization, and headline effectiveness

Each dimension contains 4-5 specific criteria scored on a 5-point scale, with detailed rubrics ensuring consistent evaluation. To ensure objective assessment, we implemented a two-stage evaluation process:

(1) **Initial scoring**: An LLM-based evaluation system trained on expert-annotated examples scored all analyses
(2) **Human verification**: Domain experts manually verified a 30% sample of these evaluations to validate scoring consistency

The human verification showed high agreement with the automated scoring, confirming the reliability of our evaluation approach.

### 3.2 Comparative Evaluation

We evaluated RADAR against three benchmarks:

(1) **Human expert analyses**: 10 analyses from supply chain specialists
(2) **Single-agent LLM**: Our initial monolithic design
(3) **Standard prompt approach**: A baseline using simplified prompting without specialized agents or preprocessing

These results demonstrate the significant advantages of our multi-agent approach over both human analyses and simpler LLM implementations. Notably, while the single-agent approach outperformed human experts in overall quality, it still fell substantially short of the multi-agent design in all dimensions.

### 3.3 Key Performance Indicators

Our evaluation revealed RADAR's strengths in several areas:

- **Root Cause Identification Accuracy**: 91% alignment with expert-verified causes, compared to 64% for single-agent and 42% for standard prompt approach
- **Data Traceability**: 98% of RADAR's claims were directly traceable to source data, compared to 37% for human analyses, 82% for single-agent, and 29% for standard prompt
- **Hallucination Rate**: 3% in RADAR, compared to 11% in single-agent and 42% in standard prompt approach (measured by claims with no data support)
- **Time Efficiency**: RADAR reduced analysis time from 8-40 hours (human) to approximately 2 hours, an 85-95% improvement

### 3.4 Generalizability Beyond Supply Chain

Our structured data processing techniques and multi-agent architecture generalize beyond supply chain to domains with similar characteristics: time-series analysis (financial markets, sensor data), multi-entity relationships (CRM, healthcare records), and multi-factor causality (fraud detection, quality control).

### 3.5 Key Insights and Technical Lessons

Key insights for LLM applications to structured data include: (1) Explicit structure preservation improved reasoning quality by 31% versus flexible formats; (2) Context-aware data summarization enhanced analytical accuracy by 28%; (3) Domain-specialized agents outperformed general-purpose LLMs by 49%; and (4) Our optimal balance shifted 65% of calculations to preprocessors while preserving LLM reasoning flexibility. (5) Systematic self-diagnosis using backward tracing reduced debugging time by multiple fold while improving fix precision. See Appendix A for additional principles.

## 4 Conclusion

RADAR demonstrates that multi-agent LLM systems with appropriate preprocessing can effectively automate complex root cause analysis of structured supply chain data. Our architecture addresses critical challenges in applying foundation models to structured data, including context length limitations, domain-specific reasoning, and hallucination management, with substantial improvements over both human experts and simpler LLM implementations.

Beyond supply chain applications, our work contributes technical approaches to structured data processing for LLMs that generalize to other domains. The systematic principles we've developed for balancing deterministic preprocessing with LLM reasoning provide a framework for effective foundation model applications across industries.

Looking forward, we plan to leverage RADAR-generated synthetic analyses for supervised fine-tuning and RLAIF. Our vision includes domain-specific fine-tuned models that combine structured reasoning with flexibility for novel scenarios, transforming analytical tasks involving structured data.

## References

[1] Judit Nagy and Pavel Foltin. 2022. Use of Big Data Analysis to identify possible sources of Supply Chain disruption through the DOTMLPFI method. *LogForum* 18, 3 (2022), 309–319.

[2] Olivier Serrat and Olivier Serrat. 2017. The five whys technique. *Knowledge solutions: Tools, methods, and approaches to drive organizational performance* (2017), 307–310.

[3] Shenao Wang, Yanjie Zhao, Xinyi Hou, and Haoyu Wang. 2024. Large language model supply chain: A research agenda. *ACM Transactions on Software Engineering and Methodology* (2024).

[4] Bin Zhou, Xinyu Li, Tianyuan Liu, Kaizhou Xu, Wei Liu, and Jinsong Bao. 2024. CausalKGPT: Industrial structure causal knowledge-enhanced large language model for cause analysis of quality problems in aerospace product manufacturing. *Advanced Engineering Informatics* 59 (2024), 102333.

## A Engineering Principles for LLM-based Structured Data Analysis

Through the development of RADAR, we established core engineering principles specifically tailored for LLM-based structured data analysis systems. These principles guided our technical decisions and can be applied to similar foundation model applications across domains.

### A.1 Architectural Design Principles

**Single Responsibility (Component Specialization)**: Each agent focuses on specific domain expertise with clearly defined boundaries. This principle enabled fault isolation, where issues in one agent (e.g., hallucinations in demand forecasting) did not cascade to other analyses. In practice, modifications to the Economics Agent improved its performance by 37% without requiring changes to other components.

**Multi-Agent Coordination**: We established standardized information exchange protocols between agents, explicit responsibility boundaries, and synthesis prioritization rules. Our synthesis agent uses weighted evidence assessment based on data quality markers from each specialized agent, particularly valuable when agents produced conflicting interpretations.

**Progressive Computational Offloading**: As we identified reliable patterns, we progressively moved calculations from LLMs to preprocessors. For example, shifting product substitutability calculations from LLMs to preprocessing code reduced hallucinations by 28% and improved computational efficiency by 48%, while preserving the LLM's reasoning capabilities for explaining the implications of these relationships.

### A.2 Data Processing Principles

**Input Data Quality and Structure**: Through experimental comparison, we found that providing LLMs with hierarchically structured data improved analysis quality by 39% over flat data formats. Our optimal approach organizes data by relevance hierarchy, with critical KPIs positioned prominently in the context window while maintaining access to supporting details.

**Data Consistency Across Components**: We implemented shared metric definitions and standardized time series interpretations across all agents. This standardization reduced contradictory findings by 65% compared to our earlier implementations where each agent used slightly different metric definitions.

**Fact vs. Opinion Separation**: We explicitly distinguished between data-driven facts and interpretive analysis through structured templates. Each conclusion references specific evidence with confidence levels, reducing ungrounded claims by 72% compared to earlier versions.

### A.3 Agent Design Principles

**Explicit Memory Management**: We found that LLM agents without explicit memory mechanisms tend to lose track of complex reasoning chains. Our implementation of structured logging mechanisms that capture intermediate conclusions improved consistency by 47% across multi-step analyses.

**Tool Augmentation Over Prompt Expansion**: When agents needed to perform complex calculations, we found providing purpose-built functions outperformed adding detailed calculation instructions to prompts. This function-calling approach reduced token consumption by 61% while improving calculation accuracy by 38%.

**Data-Grounded Response Protocol**: We implemented strict protocols requiring agents to explicitly indicate when information is unavailable rather than attempting extrapolation. This "honest uncertainty" principle reduced hallucination rates by 76% compared to early prototypes where agents attempted to provide complete answers regardless of data limitations.

**Modular Prompt Engineering**: By structuring prompts as functional modules (system context, task definition, reasoning framework, response formatting), we enabled targeted improvements without disrupting entire agent behaviors. This approach reduced regression issues by 83% compared to holistic prompt updates.

### A.4 System Improvement Principles

**Extension Without Modification**: When adding new capabilities, we extended rather than rewrote existing prompts. This approach preserved previously validated functionality while incrementally improving system capabilities, reducing regression issues by 81% compared to full prompt rewrites.

**Systematic Backward Tracing**: Our debugging methodology always works backward from outputs to inputs, verifying information presence at each stage. This approach identified that 68% of quality issues originated in preprocessing while 24% came from agent instructions and only 8% from synthesis.

**Human-in-the-Loop Integration**: We systematically incorporated expert feedback through structured prompt updates. Evaluations showed this approach accelerated system improvement by 3.2x compared to purely algorithmic optimization.

**Quantitative Performance Management**: We continuously monitored token usage and processing time, optimizing the balance between preprocessing and LLM computation. This allowed us to maintain consistent performance while processing increasingly complex analyses.

These principles form a comprehensive framework for developing LLM-based structured data analysis systems that deliver consistent, high-quality results while enabling continuous improvement. By applying these principles systematically, we achieved substantial gains in analysis quality, system reliability, and development efficiency.