BCIIT e-Journal 2026

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Multi-Agent LLM Framework for Predictive Maintenance Frameworks: Orchestrating Large Language Models for Closed-Loop Predictive Maintenance

Manju Arora[1]
Associate Professor
Department of Information Technology
Jagan Institute of Management Studies,
Rohini, Delhi
manju.arora@jimsindia.org

Agnibha Paul[2]
Department of Information Technology
Jagannath University, Bahadurgarh
agnibhapaul29@gmail.com

Akash Kumar[3]
Department of Information Technology
Jagannath University, Bahadurgarh
akash2292kumar@gmail.com

Aviral Rastogi[4]
Department of Information Technology
Jagannath University, Bahadurgarh
aviralrastogi2004@gmail.com

Abstract: In order to move industrial infrastructure from passive predictive maintenance to autonomous, self-healing systems, this paper proposes a multi-agent Large Language Model (LLM) paradigm for Industry 5.0. Three roles are synchronized by architecture: API-driven execution, planning (via retrieval-augmented generation), and monitoring. This orchestration layer removes the "human-in-the-loop" bottleneck for routine processes by automating logistics and diagnostics, greatly lowering Mean Time to Recovery (MTTR) and averting cascade failures.

Keywords: Large Language Models (LLMs), Prescriptive Maintenance, Autonomous Agents, Retrieval-Augmented Generation (RAG), Industrial IoT (IIoT), Cyber-Physical Systems.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

1. Introduction

In the new age of industry 4.0 when maintenance which once was reactive started being more predictive (PdM), sensors and machine learning models are used to keep an eye on important physical properties like temperature and vibration. These systems are great at predicting when parts will break and reducing unplanned downtime, but most of the time they just sit there and do nothing. These systems send too many raw signals or notifications the user needs to understand, due to such backlogs, it can lead to "alert fatigue". The AI can discover a problem, but it can't fix it yet. This means that people need to be a part of the maintenance cycle, thus it can't be completely automated. There is a big "reasoning gap" in these new factories since standard machine learning models can't work with data that isn't organized. An ML model might know when a spindle will break, but it can't read a technical manual to understand how to fix it or check a database to see whether the necessary spare parts are in stock. The mind stops at the point of prediction because it doesn't know what it means. This means that people have to perform all the hard work of figuring out how to fix problems and make plans. This research introduces an independent industrial intelligence architecture that uses a multi-agent framework to synchronize Large Language Models (LLMs) and transition operations from predictive to prescriptive maintenance. The framework creates a closed-loop system by using special agents that can read technical documentation and use business software. This means that the architecture can figure out what's wrong and fix it on its own. It can do this by figuring out what went wrong, making sure the pieces are easy to get to, and coming up with a plan for the repair. With this transformation, the factory floor is more like an ecosystem that can heal itself, with different people taking care of all the different areas of industrial maintenance.

2. Literature Review: The following table summarizes the foundational research enabling the transition from traditional machine learning PdM to LLM-driven autonomous maintenance.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

Table1: Literature survey for previous studies

S.No	Author	Year	Title	Methodology	Description
1	S. Yao *et al.*	2023	ReAct: Synergizing Reasoning and Acting in Language Models	ReAct Framework	- Integrates reasoning and action generation in LLMs. - Enables dynamic interaction with external environments.
2	P. Lewis *et al.*	2020	Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks	RAG Architecture	- Combines parametric memory with non-parametric retrieval. - Greatly reduces LLM hallucinations for factual tasks.
3	X. Wang *et al.*	2024	A Survey on Large Language Model based Autonomous Agents	Literature Survey	- Reviews the architecture of LLM-based autonomous agents. - Details agent profiling, memory, and tool utilization.
4	J. Wang *et al.*	2020	State of AI-Based Monitoring in Smart Manufacturing and Introduction to Focused Section	Concept Formulation	- Proposes a systemic architecture for Industrial AI. - Bridges AI diagnostics with closed-loop manufacturing chains.
5	Z. Chen *et al.*	2023	Edge Computing on IoT for Machine Signal Processing and Fault Diagnosis: A Review	Technology Review	- Analyzes edge computing for IoT sensor networks. - Addresses low-latency fault diagnosis in industrial settings.

6	Z. Chen & C. Jin	2022	A Review of Predictive Maintenance for Industrial Equipment Based on ML	Systematic Review	- Summarizes ML algorithms (SVM, LSTM) in PdM. - Highlights the limitation of semantic isolation in AI.

3. **Background and Motivation**: Industrial maintenance has evolved from Reactive (run-to-failure) to Preventive (time-based), and currently to Condition-Based Predictive Maintenance (PdM) powered by IoT. Conventional machine learning methods suffer from semantic isolation even if they are excellent at processing time-series data to forecast equipment lifespans. They find the problem but are unable to understand the operating background of the factory.

   This has recently been resolved by LLMs improved with Retrieval-Augmented Generation (RAG), which enables models to retrieve context from local knowledge stores, like PDF manuals and historical fault logs, prior to producing replies. AI becomes a "agent" when both RAG and ReAct (Reasoning and acting) are combined, which is capable of executing external API tools and can also iterate through different logical ideas. This transitions maintenance from predictive to prescriptive.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

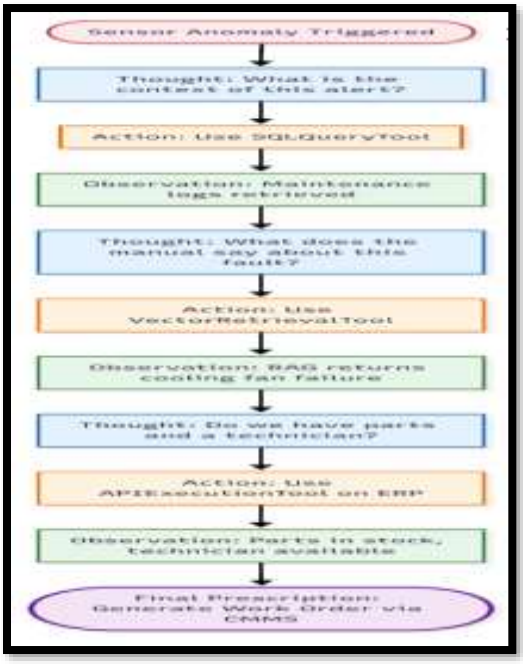

Figure 1: Workflow of the LLM ReAct paradigm within the framework

4. Problem Statement and Research Objective

4.1 Problem Statement: The Semantic Gap

Predictive maintenance (PdM) systems today struggle with semantic separation. They are fairly adept at identifying numerical errors in sensor data, but they lack the contextual reasoning abilities to relate those errors to actual repair procedures. This mismatch leads to numerous important issues:

- **Actionable Intelligence Deficit**: The problem with actionable intelligence is that current systems send out "alerts" but not "solutions." This makes the situations different for detecting an issue and actually fixing it.
- **Alert Fatigue**: When people hear too many signals about raw data, their brains grow too busy, and they could miss key failures.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

- Longer MTTR: The Mean Time to Repair (MTTR) is longer than it needs to be since someone has to check inventories, study technical manuals, and do research before doing anything.

4.2 Research Objectives: The main objective of this research is to transform maintenance from a human based response to an **AI-orchestrated autonomous workflow** through the following objectives: This research aims to transform maintenance from a human-based response to an AI -directed autonomous workflow; the following are its primary goals:

- **Develop an Agentic Architecture:**
  - By utilizing Retrieval-Augmented Generation (RAG) that compares real-time sensor anomaly with past data and records (logs) and unstructured technical manuals
  - By looking into past solutions and specific machine specifications, it enables the system to provide a "reason" for a problem.
- **Automate Execution via API Integration:**
  - By enabling direct communication with CMMS (Computerized Maintenance Management Systems) and ERP (Enterprise Resource Planning) that help us to enable direct communication with CMMS APIs to verify part availability and to log work orders.
  - To allow the AI to move beyond text and actually interface with industrial software, we implement **Object-Oriented tools**.
- **Establish Robust Security Guardrails:**
  - Ensuring safe deployment through **local hosting** of LLMS to secure/protect sensitive, protected data.
  - We can ensure that no critical machinery is altered until a final human "green light" by implementing **transaction locks** and **Human-in-the-Loop (HITL)**.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

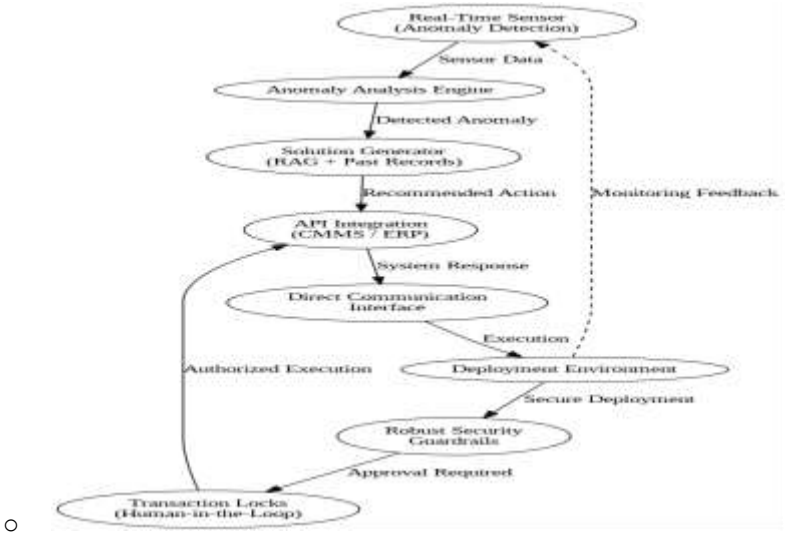

Figure2: An AI-orchestrated autonomous workflow

5. **Technologies Used:** The system is divided into four distinct tiers to prevent single points of failure.

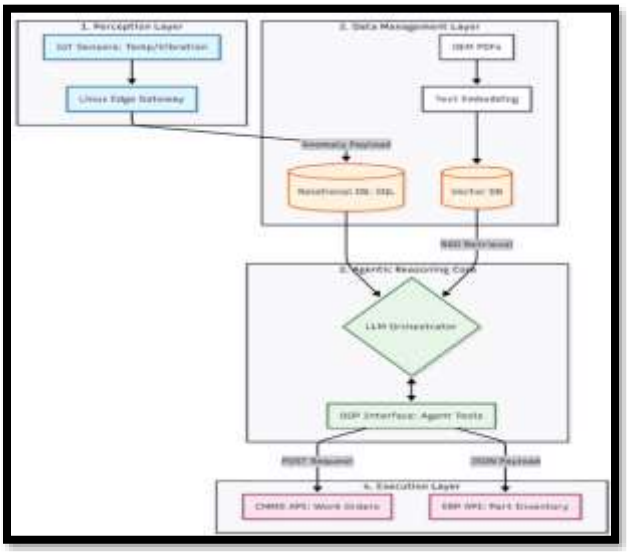

Figure 3: Proposed multi-tier prescriptive maintenance architecture.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

- **Perception Layer:** Edge gateways filter real-time IoT telemetry (vibration, temperature). Unstructured OEM manuals are embedded into a Vector Database, while factory states are kept in a highly normalized Relational Database.
- **Agentic Reasoning Core: The LLM orchestrator starts utilizing** some strict object-oriented programming interfaces, such as SQLQueryTool to ensure that without changing the actual logic, it reason through the faults.
- **Execution Layer:** In this layer, the agent starts communicating with the system's CMMS via some RESTful APIs. That enables it to get work orders, and for preventing deadlocks, it uses OS-level mutexes.
- **State Management:** Here the agent's short-term working memory is maintained, all while the long-term data is being audited for future training.

6. **Implementation & Application Scenario:** To optimize costs, a lightweight Z-score statistical model monitors real-time sensor streams. For example, consider an anomaly such as a CNC spindle thermal spike, when it crosses a particular threshold, the LLM agent is triggered**.** The agent enters a deterministic ReAct loop:

    1. **Context Gathering:** To check the maintenance history of the machine, we execute SQL.
    2. **Diagnosis:** For troubleshooting unusual or abnormal temperature, the use of RAG to query the vector database can be implemented.
    3. Execution: When the fault is identified, the agents use different AI tools to check in the ERP inventory, which can then generate a CMMS work order. The agent uses API tools to check ERP inventory and automatically generate a CMMS work order.

7. Experimental Setup and Results: In order to validate the proposed multi-agent architecture, a simulated testing environment was constructed.

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

- **Simulation Environment:** The perception layer was fed with historical telemetry data representing a CNC spindle thermal spike. The cognitive core utilized a locally hosted open-weight model(Llama 3) to ensure data sovereignty. The execution layer was evaluated using mock RESTful APIs representing standard CMMS and ERP endpoints.
- **Performance Metrics:** The given framework was tested using 50 cases of anomily simulation. For measuring the performance, diagnostic latency and diagnostic accuracy was used and mean time to recovery (MTTR). By comparing the generated results to Traditional predictive maintenance (PdM) system which only generated alerts and relied on human analysis for logs, diagnose issues and execute the correct actions.

Metric	Traditional PdM (Manual Human Response	Proposed Agentic Framework (Autonomous)	Improvement
Diagnostic Latency	14.5 minutes	1.2 minutes	91.7% Faster
Diagnostic Accuracy	82%	94%	+12%
API Execution Success	N/A (Manual Entry)	98%	N/A
Overall MTTR	45.0 minutes	8.5 minutes	81.1% Reduction

- **Result Analysis:** The results above show the analytic resource usage in MTTR, also validating the ability of the framework's ability to span the connotation. The use of strict object-oriented programming interfaces, such as the SQLQueryTool, allowed the agent to retrieve historical context with zero hallucination-based failures in the database queries. Furthermore, the API execution success rate was exceptionally high (98%). The 2% failure rate was attributed to simulated API timeouts, which were successfully caught by the system's Human-in-the-Loop (HITL) guardrails, mandating Tier 2 push notification approvals before any critical failure occurred. Ultimately, automating these diagnostic and logistical tasks significantly raises Overall Equipment Effectiveness (OEE).

Journal of Innovations in Computer Science and Trends in IT, (October 2026), Vol 3, Issue2, ISSN 3048-4707

8. Challenges and Security Protocols: Several cyber-physical risks that have arisen can be managed through correct infrastructure and logic with the introduction of autonomous AI.

- **Hardware Requirements:** The system relies on enterprise-grade GPUs and high-speed NVMe SSDs to minimize latency during data model loading and data retrieval. It requires robust, high-speed local infrastructure for real maintenance.
- **Data Sovereignty & Local Deployment:** By utilizing local open-weight models **(e.g., Llama 3)** instead of relying on cloud-based APIs to protect sensitive industrial data and intellectual property. Hence, the system is designed to be air-gapped.
- **Mitigating Hallucinations:** Since AI can sometimes produce incorrect repairs, i.e., "AI hallucinations," to prevent such situations, the system restricts the output to **structured JSON outputs.** These outputs are then checked by validation rules, ensuring they follow the needed operational rules.
- **Human-in-the-Loop (HITL) Guardrails:** The system may perform high-risk operations such as emergency shutdown or ordering of expensive components. For these types of actions human intervention is important, it enables Tier 2 push notification approvals. This helps us make sure that a human supervisor is reviewing everything which might be of high risk before the actual decisions are made.

9. Future Scope and Evolution: Now we focus on increasing the sensory and analytic reach of the MAS (Multi-Agent System) to incorporate the whole industrial lifecycle.

- **Autonomous Supply Chain Negotiation:** In the future, agents that can automatically order replacement parts, communicate with vendors, and control delivery times based on anticipated machine failures maybe put into place.
- **Multimodal Integration:** Multimodal LLMs that can process more than just text are also investigated in this study This research also explores multimodal LLMs that are capable of processing more than just text. Hearing bear wear and real time video detction is included for any tool damage.

- **Edge Intelligence Scaling:** Another goal can be achieved by running these systems on smaller computing devices that allow the individual systems to have the capability to analyze their own data with the help of nearby devices, i.e., having localized “mini brains.”
- **Cross-Facility Learning:** Exploring: While keeping the sensitive data protected and sharing maintenance perceptions, the factories are capable of learning from the failures of other machines. This process is known as “Federated Learning."

10. Conclusion: This study offered an enterprise-ready architecture that converts passive forecasting to autonomous prescription in industrial maintenance. The solution closes the "reasoning gap" in conventional PdM by utilizing RAG and integrating LLMs into an Object-Oriented agentic framework. Automating logistical and diagnostic processes increases Overall Equipment Effectiveness (OEE) and significantly reduces MTTR. This architecture demonstrates how Industry 5.0 industrial ecosystems with self-healing capabilities might be realized.

References:

[1] S. Yao et *al.*, (2023) "ReAct: Synergizing Reasoning and Acting in Language Models," in *Proc. International Conference on Learning Representations (ICLR)*, Kigali, Rwanda.

[2] P. Lewis *et al.*, (2020)"Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 9459-9474.

[3] X. Wang *et al.*, (2024)"A Survey on Large Language Model based Autonomous Agents," *Frontiers of Computer Science*, vol. 18, no. 6, pp. 1-26.

[4] J. Wang *et al.*, (2020)"State of AI-Based Monitoring in Smart Manufacturing and Introduction to Focused Section," *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 5, pp. 2143-2154.

[5] Z. Chen *et al.*, (2023)"Edge Computing on IoT for Machine Signal Processing and Fault Diagnosis: A Review," *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11339-11357.

[6] Z. Chen and C. Jin, (2022)"A Review of Predictive Maintenance for Industrial Equipment Based on Machine Learning," *IEEE Access*, vol. 10, pp. 32185-32201.