computers

IMPACT FACTOR 4.2

CITESCORE 7.5

Article

Artificial Intelligence Agent-Enabled Predictive Maintenance: Conceptual Proposal and Basic Framework

Wenyu Jiang and Fuwen Hu

Special Issue

Adaptive Decision Making Across Industries with AI and Machine Learning: Frameworks, Challenges, and Innovations

Edited by
Dr. Samiul Islam

MDPI

https://doi.org/10.3390/computers14080329

computers

MDPI

Article

Artificial Intelligence Agent-Enabled Predictive Maintenance: Conceptual Proposal and Basic Framework

Wenyu Jiang and Fuwen Hu *

Brunel London School, North China University of Technology, Beijing 100144, China; jiangwenyu123@mail.ncut.edu.cn
* Correspondence: hfw@ncut.edu.cn

Abstract

Predictive maintenance (PdM) represents a significant evolution in maintenance strategies. However, challenges such as system integration complexity, data quality, and data availability are intricately intertwined, collectively impacting the successful deployment of PdM systems. Recently, large model-based agents, or agentic artificial intelligence (AI), have evolved from simple task automation to active problem-solving and strategic decision-making. As such, we propose an AI agent-enabled PdM method that leverages an agentic AI development platform to streamline the development of a multimodal data-based fault detection agent, a RAG (retrieval-augmented generation)-based fault classification agent, a large model-based fault diagnosis agent, and a digital twin-based fault handling simulation agent. This approach breaks through the limitations of traditional PdM, which relies heavily on single models. This combination of "AI workflow + large reasoning models + operational knowledge base + digital twin" integrates the concepts of BaaS (backend as a service) and LLMOps (large language model operations), constructing an end-to-end intelligent closed loop from data perception to decision execution. Furthermore, a tentative prototype is demonstrated to show the technology stack and the system integration methods of the agentic AI-based PdM.

Keywords: predictive maintenance; agentic artificial intelligence; large language models; large language model operations

Academic Editors: Samiul Islam and Leandros Maglaras

Received: 16 July 2025
Revised: 9 August 2025
Accepted: 12 August 2025
Published: 15 August 2025

Citation: Jiang, W.; Hu, F. Artificial Intelligence Agent-Enabled Predictive Maintenance: Conceptual Proposal and Basic Framework. *Computers* **2025**, *14*, 329. https://doi.org/10.3390/computers14080329

1. Introduction

Basically, maintenance approaches are categorized into three foundational types: reactive, preventive, and predictive [1,2]. Condition-based maintenance (CBM; for more abbreviations in the text, see the list of abbreviations at the end) and reliability-centered maintenance (RCM) represent advanced evolutions of predictive maintenance (PdM), where CBM focuses on real-time health indicators, while RCM prioritizes critical components via reliability analysis. Reactive maintenance (or run-to-failure) operates passively, addressing issues solely after equipment breakdowns occur, invariably causing unplanned downtime and emergency repairs [3,4]. Conversely, preventive maintenance (PM) adheres strictly to fixed time/usage schedules, performing routine servicing irrespective of actual equipment condition, often resulting in unnecessary upkeep. PdM leverages big data and artificial intelligence (AI)-driven analytics—synthesizing equipment process data, mathematical models, and mechanical insights—to forecast failures proactively, enabling precisely timed interventions based on condition monitoring. Briefly, RCM optimizes critical asset management; PdM and CBM use data-driven intelligence to minimize downtime; PM

offers predictability at the risk of over-maintenance; and reactive maintenance remains a costly last-resort strategy [5,6].

In fact, PdM presents considerable challenges that often hinder its successful implementation, with many solutions achieving less accuracy or falling far short of expectations. One primary obstacle is the interdisciplinary complexity and talent gap, and effective PdM necessitates a blend of data science, AI, the Internet of Things (IoT), engineering physics, and specialized domain knowledge. Unfortunately, few organizations possess this cross-functional expertise, leading to misaligned solutions that overlook real-world operational constraints and fail to integrate adequately between data teams and field engineers [7,8]. Moreover, the high initial investment required for sensors, infrastructure upgrades, and specialized software creates financial barriers, particularly when coupled with long payback periods and difficulty in quantifying return on investment (ROI) [9,10]. Data quality and accessibility also pose significant challenges, as historical fault data is often scarce or incomplete, while legacy equipment may lack adequate sensor systems. Additionally, issues such as poor sensor calibration and environmental noise can produce "dirty data" that misguides predictive models. The disconnect between historical model training and live operational environments further complicates matters, as these models can quickly degrade due to factors like equipment aging and material variability [10,11]. Furthermore, insufficient integration of domain knowledge often leads to misunderstandings of equipment failure mechanisms, exacerbated by staff turnover and an inadequate encoding of physical laws into models. Operational fragmentation occurs when PdM initiatives are isolated from maintenance workflows, reducing their effectiveness. Ultimately, the success of PdM relies on addressing these multifaceted challenges through continuous investment, fostering cross-departmental collaboration, and focusing on both technical robustness and organizational readiness [12].

The dynamic development of industrial systems in recent years has been increasingly shaped by the transition from Industry 4.0, characterized by digitalization, automation, and connectivity, to the emerging paradigm of Industry 5.0 [13,14]. Recent developments in the Maintenance 5.0 concept have created implementation challenges and possibilities. Maintenance 4.0 represents the digital evolution of maintenance within Industry 4.0, leveraging technologies like IoT, AI, and digital twin (DT) to enable predictive and autonomous decision-making. It focuses on real-time data analytics, failure prediction, and seamless integration with cyber–physical production systems to optimize asset performance and minimize downtime. Operators primarily supervise automated processes, with limited emphasis on sustainability or human factors. While significantly improving efficiency, this approach remains technology-centric and reactive to predefined risks rather than systemic disruptions [15,16]. Maintenance 5.0 transcends its predecessor by aligning with Industry 5.0 pillars: sustainability, resilience, and human-centricity [17]. It transforms maintenance into a strategic function that balances technical performance with environmental stewardship (e.g., circular economy practices) and operator well-being (e.g., virtual reality (VR)-guided collaboration) [18]. Enabled by human–AI teams and adaptive technologies like cobots, it prioritizes system recoverability from disruptions, ethical governance, and long-term value creation. This paradigm shifts operators from supervisors to co-designers, embedding resilience engineering and triple-bottom-line accountability into maintenance workflows.

Moreover, the rapid development of large models (LMs), including large language models (LLMs) and multimodal vision–language models (VLMs), has led to transformative changes across various industries [19]. Unlike traditional machine learning methods, LLMs can effectively integrate multimodal data, including structured sensor data and unstructured maintenance logs, technical manuals, and operational records [20]. This

integration capability enables LLMs to identify failure patterns that traditional machine learning methods often overlook. For instance, by correlating compressor vibration data with technician annotations on chemical exposure, LLMs can recognize degradation mechanisms undetectable by numerical models alone. Moreover, LLMs demonstrate exceptional adaptability to specific operational environments, allowing them to learn from domain-specific historical data. In addition to these inherent advantages, LLMs can also be used as the cognitive core of an agent system, connecting modular components such as memory, planning, a toolkit, and an executable workflow to expand its applicability [21]. However, LLMs are prone to hallucinations, produce seemingly reasonable but incorrect output, and often misunderstand users' intentions, which is unacceptable in the field of PdM. In terms of technical implementation, retrieval-augmented generation (RAG) significantly enhances LLM performance in industrial settings [22]. By integrating LLMs with domain-specific knowledge bases, RAG enables models to access up-to-date technical documentation, maintenance records, and operational procedures, thereby delivering more accurate and contextually relevant predictions. This approach is particularly suitable for data-sparse industrial applications, where models can semantically enrich input sequence data to enhance collaborative decision-making capabilities with plant operators [23].

With the rapid progress of LMs, the development of general-purpose intelligent agents powered by them has become achievable. AI agents, also referred to as agentic AI, constitute autonomous entities endowed with the capabilities to comprehend and respond to human inputs, perceive their environment, formulate decisions, and execute actions within diverse settings—spanning physical, virtual, or mixed-reality domains—to accomplish predefined objectives [24]. They come in various forms, ranging from virtual assistants to sophisticated multi-agent systems that coordinate logistics or financial operations [25]. This evolution signifies a significant shift in how organizations leverage AI to enhance productivity and improve human–machine interactions, marking a pivotal moment in the ongoing digital transformation. Briefly, the core architecture of LM-based agents comprises five synergistic modules: the planning module decomposes tasks and generates strategies; the memory module manages short-term context and long-term knowledge storage; the action module executes embodied operations or tool invocations; the interaction module facilitates collaboration among agents, humans, and environments; and the security module ensures privacy, threat resilience, and ethical compliance. Through integrated data flows and feedback loops, these modules enable autonomous, goal-directed operation across physical, virtual, or mixed-reality settings while enhancing system reliability and security [26,27].

In particular, no-code/low-code platforms specializing in AI application development (like Coze and Dify) and visual workflow automation platforms with AI capabilities (like n8n) empower non-developers to build, customize, and deploy AI-powered applications. This includes creating chatbots, RAG-powered assistants, and agent-like workflows by leveraging LLMs through intuitive interfaces, pre-built templates, and simplified configuration of components like data sources and actions. Inspired by the rapid progress of LMs and the large language model operations (LLMOps), this work aims to propose the method of agentic AI-based PdM. The basic contributions include the following:

First, the basic reference architecture of agentic AI-based PdM is proposed. Reference architectures help to capture the essence of existing standards, norms, and conceptual framework models and the vision of future needs and evolution to provide guidance to assist in developing new systems of agentic AI-based PdM.

Second, an exploratory implementation roadmap of agentic AI-based PdM is demonstrated. The implementation roadmap innovatively integrates AI agents, diagnosis knowledge graphs, large model reasoning, and DTs into a core framework for PdM, breaking through the limitations of traditional PdM's reliance on single models. For optimizing

bandwidth, resilience, and responsiveness of the complex PdM system, the edge–fog–cloud architecture is prioritized by tiering resources: edge nodes handle real-time, low-latency tasks; fog nodes process zonal data aggregation and mid-tier analytics; and the cloud manages global-scale computation and storage. This combination of "multi-agent (collaboration of data, models, and simulation) + RAG (diagnosis knowledge base) + reasoning engine (LMs) + LLM-powered applications (agentic AI builder)" constructs an end-to-end intelligent closed loop from data perception to decision execution.

2. AI Agent Development

2.1. AI Agent Pyramid

AI agent construction platforms are specialized subsets of LLM application development frameworks that enable developers to build, deploy, and manage goal-oriented, autonomous AI systems. These platforms abstract low-level complexities (e.g., tool orchestration, memory management) while providing modular components for agent design. Unlike generic LLM application builders, they emphasize autonomy, multi-step reasoning, and environmental interaction [28]. LLMOps extends MLOps (machine learning operations) principles to specialize in the operational management of LLM-centric systems. It first handles the unique LLM challenges: hallucination mitigation, prompt versioning, RAG pipeline monitoring, and cost-efficient scaling of stateless APIs (application programming interfaces). Second, it streamlines the lifecycle stages: data curation → prompt engineering/fine-tuning → deployment → evaluation (e.g., benchmark-driven validation) → monitoring (latency, drift, ethics) → continuous iteration [29]. MLOps, in contrast, focuses on end-to-end lifecycle management for traditional machine learning models (e.g., random forests and convolutional neural networks), prioritizing reproducibility, A/B testing, and drift detection in statistical patterns.

Figure 1 classifies agent development into three hierarchical tiers based on technical complexity. At the foundational level (L1, platform-driven), developers utilize no-code/low-code platforms such as Dify or Coze for rapid deployment, trading minimal coding requirements for constrained customization. Progressing to the intermediate tier (L2, API-driven), practitioners integrate LLM APIs (e.g., OpenAI and Anthropic) directly, achieving moderate control at the cost of limited built-in orchestration capabilities. The apex tier (L3, custom-built) implements full-stack agent frameworks like LangChain or AutoGen, offering maximum flexibility while incurring high DevOps/MLOps overhead. Each ascending tier demands progressively greater technical expertise but enables deeper alignment with complex requirements such as real-time analytics and enterprise-grade security.

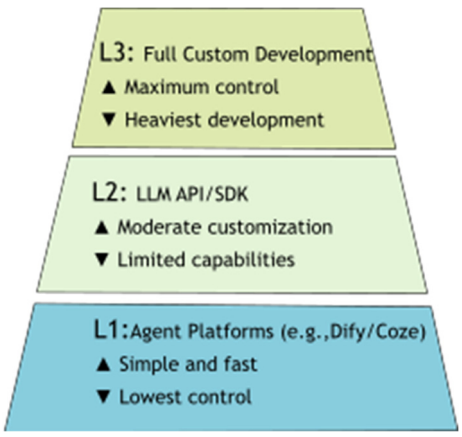

Figure 1. AI agent applications pyramid.

2.2. AI Agent Development Platform

The emergence of various AI agent construction frameworks has significantly transformed how developers create and deploy intelligent agents [30]. The goal of LLMOps is to ensure the efficient, scalable, and secure use of these powerful AI models for building and running real-world applications. It involves various aspects, including model training, deployment, monitoring, updates, security, and compliance [31].

As cataloged in Table 1, the AI agent development ecosystem has evolved into three distinct technical tiers: L1 low-code platforms (Dify/Coze) accelerate deployment through visual orchestration but limit deep customization; L2 API-driven tools (OpenAI Assistants + LangSmith) enable lightweight integration with observability for rapid prototyping; and L3 full-code frameworks (LangChain/AutoGen/Haystack) support complex system development at the cost of MLOps overhead. Innovations like Haystack's hybrid retrieval refine domain-specific accuracy, while MetaGPT pioneers agent-driven software development. This evolution centers on solving core LLMOps challenges—balancing development speed, reasoning reliability, and enterprise security—transitioning agents from experimental tools to production-grade systems.

Table 1. AI agent development platforms or toolkits.

Technical Tier	Tools	Key Platforms	Core Architecture	Technical Differentiation
L1: Low-Code Platforms	• Dify	Visual orchestration engine	• Built-in LLMOps (monitoring/eval) • Enterprise RBAC/SSO integration	Rapid deployment of secure enterprise assistants
	• Coze	Plugin-based interaction design	• Multimodal extensions (image/voice) • Cross-platform bot deployment (WeChat/Discord)	Consumer-facing social agents
	• Bubble	Web application generator	• No-code API connectors • Dynamic data binding	Business workflow automation (CRM/ERP integration)
L2: API-Driven Layer	• OpenAI Assistant API	Function calling engine	• Code interpreter + file retrieval	Lightweight knowledge assistants
	• Anthropic Claude Kit	Constitutional AI constraints	• Self-correction mechanisms • Structured output control	High-risk compliance agents
	• LangSmith	LLM observability layer	• Chain call cost/latency tracing • Prompt version comparison	LLM application debugging
L3: Full-Code Frameworks	• LangChain	Modular chaining (LCEL)	• 200+ tool integration • Async chain scheduling	Complex RAG pipelines
	• AutoGen	Conversational programming	• Interruptible agent sessions • Human-in-the-loop tuning	Multi-expert collaboration
	• Haystack	Neural search framework	• Hybrid retrieval (keyword + vector) • Domain fine-tuning	High-precision domain knowledge base
	• CrewAI	Organizational modeling	• Task dependency graphs • Resource contention handling	Enterprise task decomposition
	• LangGraph	State machine engine	• Cyclic workflow checkpoints • Distributed agent coordination	Real-time supply chain tracking

Two critical dimensions can guide the AI agent development tooling choices: team expertise and solution longevity. For teams with limited AI skills, low-code L1 platforms like Dify or Coze are optimal, accelerating deployment through visual interfaces. Teams with moderate expertise branch based on project lifespan: short-term prototypes align with L2 tools (OpenAI Assistants + LangSmith for API-driven agility and observability), whereas long-term solutions warrant L3 frameworks like AutoGen or Haystack for deeper customization. Highly skilled teams default to L3 frameworks such as LangChain or CrewAI, accepting higher DevOps overhead for maximum control in complex scenarios. This structured approach balances development speed against technical flexibility—prioritizing rapid iteration for transient needs while steering sustained initiatives toward scalable, enterprise-grade architectures.

AI agents interact with external resources (databases and APIs) and other agents via standardized protocols to enable cross-platform collaboration and resolve system integration barriers. Protocol types are defined by interaction targets: context-oriented protocols (e.g., model context protocol, MCP) and inter-agent protocols (e.g., agent-to-agent, A2A). MCP provides a unified abstraction layer for agents to discover and access contextual resources (prompts, tools, and data). MCP decouples agents from underlying implementations through semantic specifications, allowing framework-agnostic integration of community-hosted MCP servers. This enables automated workflows like data scraping and cross-system API orchestration without language/framework constraints. A2A governs stateful collaboration between multiple agents for complex tasks. A2A standardizes message passing (intent/result routing), state synchronization (task progress), and joint output aggregation, supporting conversational coordination and delegated subtask execution.

3. AI Agent Architecture for Predictive Maintenance

Figure 2 illustrates an integrated AI agent architecture for industrial PdM, forming a closed-loop system from physical assets to proactive actions. The design bridges edge/cloud layers and digital twins to overcome data scarcity and model brittleness while enabling physics-informed AI.

3.1. Data Acquisition and Preprocessing

Industrial equipment (e.g., turbines and pumps) instrumented with IIoT (industrial Internet of Things) sensors streams raw operational data (vibration, temperature, and pressure) to an edge gateway. This gateway performs latency-sensitive preprocessing: noise filtering, compression, and downsampling [32,33]. Processed data feeds into a unified ingestion pipeline where feature extraction (e.g., fast Fourier transform (FFT) for vibration) and normalization occur. Crucially, this stage outputs structured telemetry synchronized with the digital twin's virtual sensors, ensuring temporal alignment between the physical and virtual worlds.

3.2. Digital Twin

The DT serves as a physics-based virtual replica that synergizes with AI through three critical mechanisms [34,35]: (1) real-time anomaly diagnosis via counterfactual simulations (e.g., modeling "bearing clearance + 0.1 mm" to pinpoint root causes by comparing simulated vs. actual sensor outputs), (2) synthetic failure data generation for training AI agents on rare faults (e.g., turbine blade crack propagation) to enable accurate remaining useful life (RUL) forecasting, and (3) closed-loop validation of maintenance actions by simulating operational impacts (downtime, energy costs) against constraints like sustainability targets. This physics–AI fusion resolves PdM's core challenges: data scarcity, model brittleness, and operational silos [36,37].

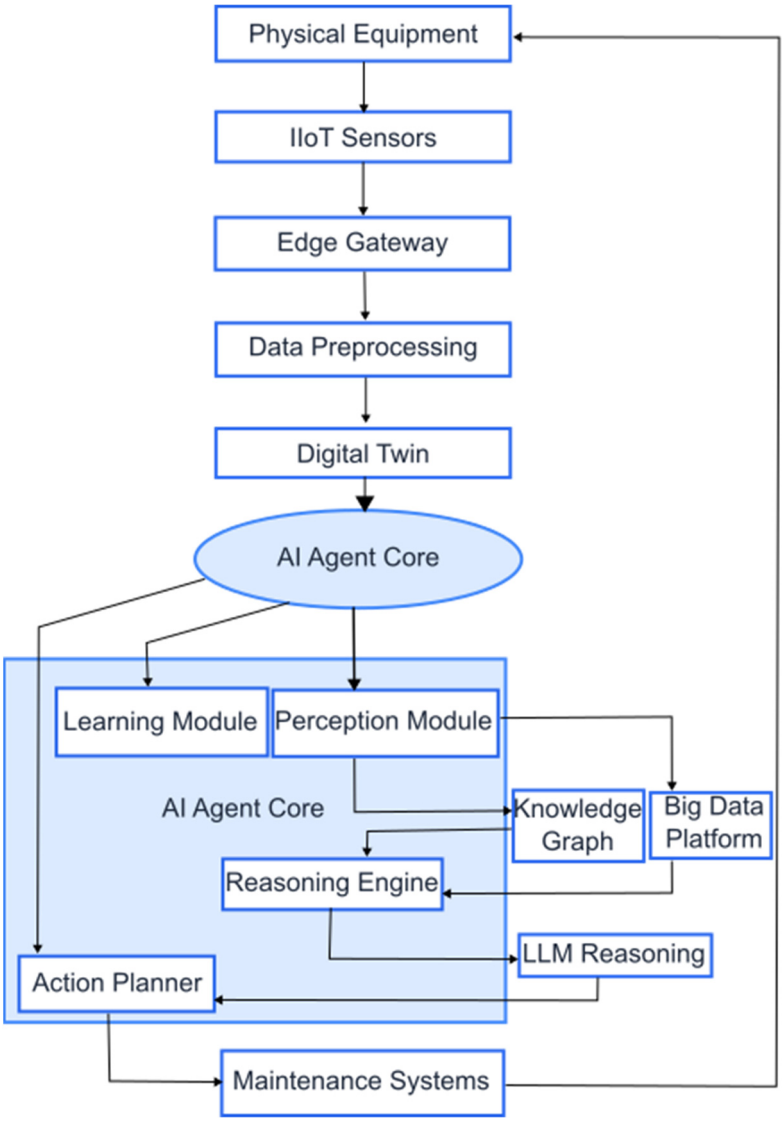

Figure 2. AI agent architecture for predictive maintenance.

3.3. AI Agent Core

Once preprocessed, the data enters the AI agent core, which is structured into four interconnected modules. The perception module integrates multi-source data (IIoT sensors, digital twin simulations, and external APIs) to construct a unified operational context for real-time monitoring. Next, as illustrated in Figure 3, the reasoning engine serves as the "cognitive hub," utilizing LLM reasoning techniques, such as DeepSeek models, to analyze data against a knowledge graph containing equipment manuals, failure histories, and physics models. This integration facilitates root cause diagnosis and failure prediction. The action planner translates insights derived from the reasoning engine into executable strategies that prioritize maintenance tasks, trigger alerts, or adjust production schedules. Lastly, the learning module acts as the "adaptive brain" of the architecture, refining the agent through reinforcement learning by incorporating feedback from maintenance outcomes, thus updating the knowledge graph and optimizing future decision-making processes [38,39].

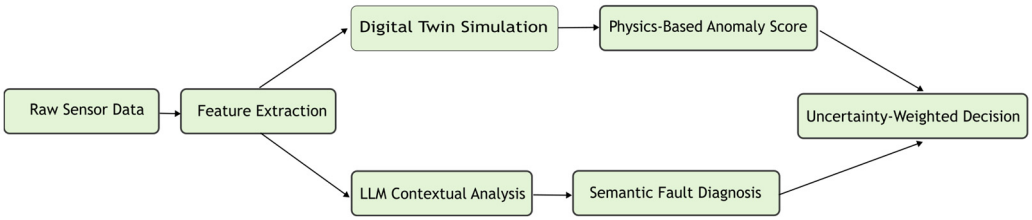

Figure 3. Reasoning engine for predictive maintenance.

3.4. Knowledge Integration

The reasoning engine interfaces with two critical knowledge repositories [40,41]. The big data platform stores historical sensor data, maintenance records, and performance logs, enabling comprehensive trend analysis. The knowledge graph, a semantic network, links equipment schematics, failure modes, and repair protocols, facilitating contextual reasoning; for example, if a bearing experiences vibration at 200 Hz combined with high temperature, it may indicate lubrication failure.

Finally, the action planner dispatches optimized instructions to various maintenance systems, which may include generating automated work orders in computerized maintenance management software (CMMS), providing augmented reality (AR)-guided repair instructions for technicians, and dynamically reallocating resources to minimize downtime. This integrated approach significantly enhances PdM strategies in industrial settings, ultimately driving improved reliability and efficiency across operations.

4. AI Agent Implementation Roadmap for Predictive Maintenance

4.1. Development Roadmap for AI Agent-Driven Predictive Maintenance System

As mentioned above, teams should honestly assess their coding proficiency, LLM lifecycle experience, and DevOps capacity before choosing frameworks. Considering the explorative and preliminary proposal of the agentic AI-based PdM method, herein the framework selection for the AI agent-enabled PdM system prioritizes Dify, RAGFlow, and DeepSeek due to their complementary strengths addressing industrial maintenance challenges. Dify (https://www.dify-china.com/ (accessed on 1 May 2025)) prioritizes ease of deployment and provides drag-and-drop assembly of chains (retrieval → reasoning → action) and basic agent-team coordination via pre-built templates. Backend as a service (BaaS) simplifies the development process by providing pre-built backend functionalities, allowing developers to focus on application features and user experience. RAGFlow (https://github.com/infiniflow/ragflow (accessed on 1 May 2025)) integrates retrieval-augmented generation with process control, dynamically fusing fragmented industrial knowledge (equipment manuals and failure histories) while mitigating AI hallucinations through real-time knowledge base access. DeepSeek (https://www.deepseek.com/ (accessed on 1 May 2025)) provides physics-based reasoning capabilities, essential for equipment failure diagnosis, by combining multimodal sensor data with physical models for accurate root cause analysis.

As illustrated in Figure 4, the proposed predictive maintenance system employs a three-tiered edge–fog–cloud architecture to achieve self-evolving capabilities. Three-phase implementation begins at the edge layer, deploying IIoT sensors and PLC gateways for real-time equipment monitoring (vibration, temperature, and current). Raw data undergoes local preprocessing via anomaly filters and feature extraction techniques. This transforms raw signals into validated, transmission-ready insights. Briefly, edge gateways perform real-time preprocessing—noise filtering, compression, and feature extraction—to minimize latency and bandwidth load. Critical alerts (e.g., threshold violations) trigger immediate actions (e.g., emergency shutdowns), while aggregated data streams to the fog layer [42,43].

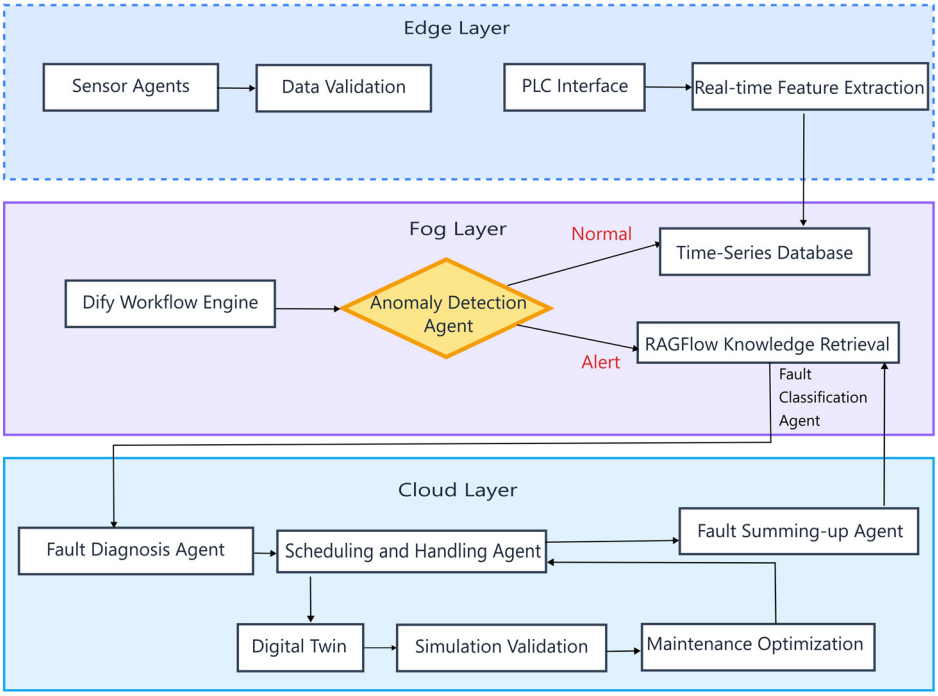

Figure 4. AI agent-driven predictive maintenance system development roadmap.

The fog tier acts as a local intelligence hub between edge and cloud. Fog nodes execute lightweight AI models for real-time anomaly detection and preliminary diagnostics, enabling rapid response without cloud dependency. They also aggregate data from multiple edge devices, reducing redundancy before transmission. Crucially, fog hosts AutoGen multi-agent planners that coordinate local maintenance schedules and resource allocation, ensuring swift decision-making for time-sensitive tasks. Processed edge data feeds into the fog layer powered by Dify, acting as the workflow engine. Here, trained classifiers detect anomalies (e.g., abnormal vibration thresholds) with 95% accuracy. Dify directs workflows via conditional branching: anomalies trigger RAGFlow for contextual diagnosis, while normal data streams to time-series databases. Reinforcement learning continuously calibrates alert thresholds to minimize false positives, avoiding unnecessary cloud processing. RAGFlow is an AI technology architecture that combines RAG with flow control. It retrieves information from external knowledge bases and utilizes natural language generation techniques to provide users with more accurate and enriched responses. Additionally, it introduces a process control mechanism that dynamically adjusts the information retrieval and generation processes based on task requirements, thereby better accomplishing complex tasks. The core advantage of RAGFlow lies in its integration of retrieval and generation, overcoming the limitations of traditional generative models that rely solely on internal parameters for text generation. It can fetch in real-time the latest and most relevant information from external knowledge bases related to user queries, then generate high-quality answers based on this information. This architecture not only enhances the accuracy and timeliness of PdM but also effectively mitigates the common issue of "hallucination" found in generative models.

The cloud layer handles computation-intensive tasks and global optimization. At the cloud layer, RAGFlow retrieves maintenance knowledge using anomaly metadata, cross-referencing manuals and failure histories. DeepSeek synthesizes sensor data, RAG context, and digital twin simulations to diagnose root causes (e.g., "impeller imbalance") and forecast remaining lifespan. The scheduling and handling planner then coordinates resources and production reality to optimize maintenance schedules, while digital twins

validate actions for downtime/cost impact before execution. Post-action feedback refines the system iteratively. Maintenance outcomes update RAGFlow's knowledge base and retrain Dify's models monthly.

In the proposed AI agent-enabled predictive maintenance framework, the system is structured around four core agents with clearly defined responsibilities: the diagnostic agent, the scheduling agent, the evaluation or summing-up agent, and the digital twin fault handling simulation agent. The diagnostic agent focuses on fault root cause analysis by integrating real-time sensor data, digital twin simulation results, and knowledge retrieved from RAGFlow. It generates structured diagnostic reports, including fault types, locations, probabilities, and RUL predictions, while also classifying anomalies into risk levels (high/medium/low) to trigger appropriate responses. The scheduling agent takes the diagnostic reports as input and optimizes maintenance tasks considering resource constraints (inventory and personnel), production schedules, and multi-objective balancing between efficiency, cost, and sustainability. It generates maintenance work orders with specific content, execution times, and assigned personnel while also dynamically adjusting priorities based on risk levels. The summing-up agent serves as the feedback loop by collecting maintenance execution data, comparing diagnostic predictions with actual outcomes, and updating the knowledge base in RAGFlow with successful cases. It also facilitates model retraining by refining DeepSeek's reasoning models and scheduling decision rules based on technician evaluations and system performance metrics. The digital twin-based agent provides simulation of the maintenance scheme and testing of the maintenance effect so as to reduce the impact on the production schedule and prevent secondary failures or hidden dangers. The collaboration between these agents follows a message-driven workflow where the diagnostic agent sends reports to the scheduling agent, which requests resource constraints from the evaluation agent before making decisions. The evaluation agent then provides feedback to the diagnostic agent after maintenance execution, creating a continuous learning loop. This architecture emphasizes clear separation of concerns, with each agent having specific responsibilities without overlap, while maintaining efficient closed-loop optimization through knowledge engineering and causal reasoning.

4.2. Prototype Demonstration

In order to make the implementation roadmap of agentic AI-based PdM more intuitive, as a proof of concept, a demo project is demonstrated, which uses the low-code visual orchestration ability of the Dify platform, the RAGFlow knowledge base component, the Deepseek reasoning ability, and the simulated equipment sensor data and enterprise resource data to build an agent AI-based PdM system for wind turbine rolling bearings. Although this project is a simple example, it clearly shows the basic skeleton, technology stack, and procedural integration of building an agentic AI-based PdM.

Figure 5 shows the basic routine to build an agentic AI-based PdM system from data acquisition to actionable insights, with parallel processing paths for real-time alerting and predictive maintenance. Figure 6 is the visual low-code workflow in the Dify platform. The prototyping process indicates that Dify reduces LLM app development from weeks to hours by abstracting LLMOps complexities while retaining extensibility via Python 3.9/API hooks.

When the developed prototype system is put into practical application, the agentic AI-based PdM system begins at the data parsing node, where industrial devices (PLCs/SCADA) transmit sensor data via OPC UA/MQTT protocols. Then, the anomaly detection node will judge if the parsed vibration signals have undergone threshold validation. If exceeded, a bearing diagnosis workflow activates, while normal data streams directly to InfluxDB for storage. The fault diagnosis workflow leverages RAGFlow to

retrieve relevant technical documentation from its knowledge base, enabling the DeepSeek model to generate root cause reports and maintenance strategies. The summing-up node serves as the feedback loop by collecting maintenance execution data, comparing diagnostic predictions with actual outcomes, and updating the knowledge base in RAGFlow with successful cases. Crucially, RAGFlow's knowledge base serves a dual role—supporting both real-time diagnostics and long-term analytics. This creates a symbiotic loop: new maintenance outcomes continuously update the knowledge repository, while historical records refine predictive models.

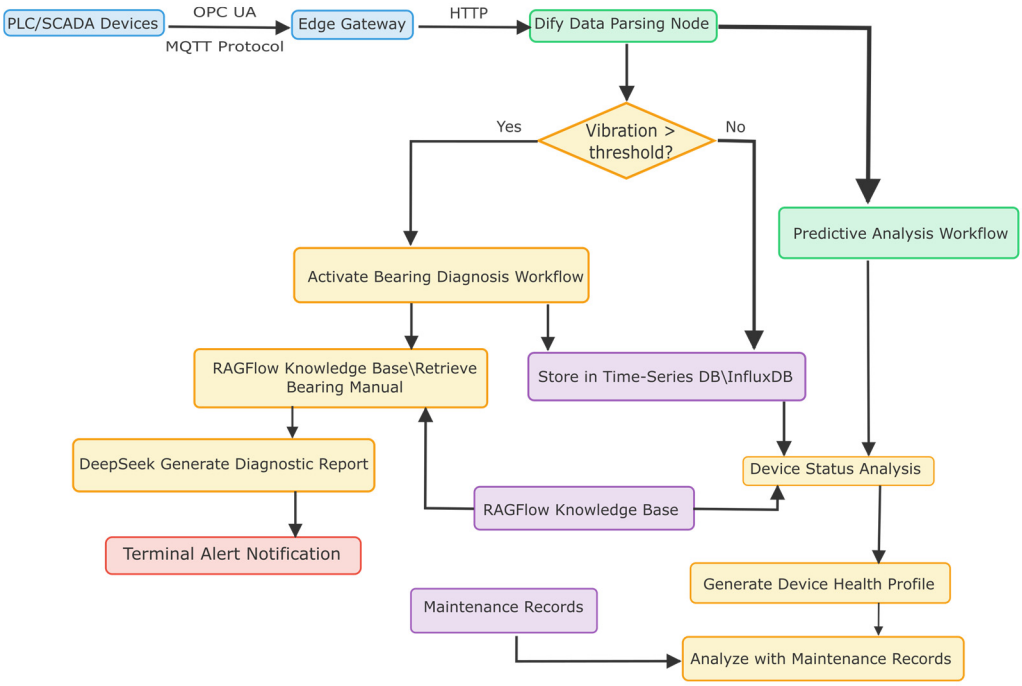

Figure 5. Basic workflow of the AI agent-driven maintenance of wind turbine rolling bearings.

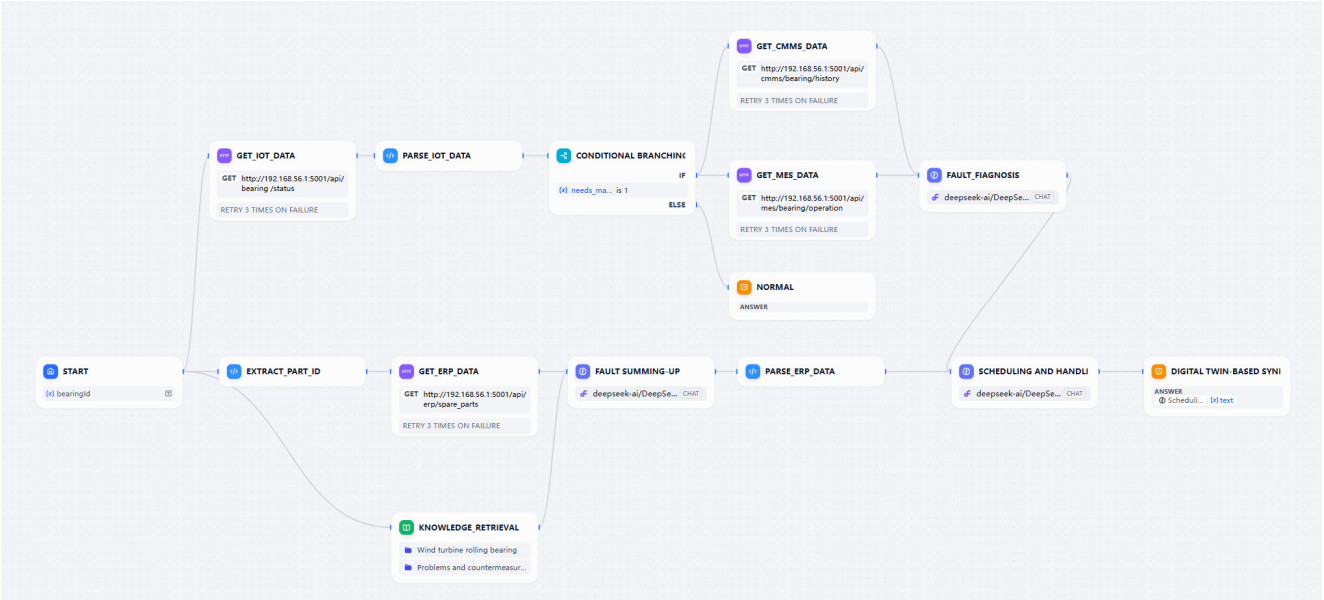

Figure 6. Visual Dify nodes of the AI agent-driven maintenance system of wind turbine rolling bearings.

4.3. Discussion

This work introduces transformative innovations in PdM by integrating multi-agent systems, RAG-enhanced knowledge bases, large model reasoning, and digital twins to overcome traditional limitations. As the contrastive analysis shown in Table 2, unlike conventional PdM architectures reliant on isolated models and static rules [44,45], our framework establishes an end-to-end intelligent closed loop from data perception to decision execution. A key advancement is the tiered AI agent development paradigm (L1/L2/L3), which democratizes PdM system development through low-code platforms while enabling enterprise-grade customization. By leveraging RAG for real-time knowledge retrieval and digital twins for risk simulation, we address hallucination issues and enhance decision reliability. Furthermore, LLMOps-driven operational mechanisms ensure continuous model refinement and feedback-driven optimization, making the system adaptive to evolving industrial environments. This holistic approach significantly advances PdM from reactive, single-model solutions to proactive, knowledge-rich, and resilient industrial intelligence systems.

Table 2. Comparison between the traditional PdM method and the proposed method.

Dimension	Traditional PdM Method	Proposed Method
Technical Architecture	Reliance on single models, static rules	Multi-agent collaboration + RAG + large models + digital twin closed-loop
Development Paradigm	Dependency on specialized development teams	Tiered agent platforms (L1/L2/L3) lowering entry barriers
Knowledge Utilization	Siloed historical data, delayed updates	RAG enables real-time retrieval of the latest knowledge bases and dynamic updates
Decision Reliability	Lack of risk simulation	Digital twins pre-test maintenance actions to avoid unexpected and hidden dangers
Operational Mechanism	Model degradation, no iterative optimization	LLMOps-driven continuous learning and feedback loops

5. Conclusions

This research pioneers an agentic AI framework for predictive maintenance that fundamentally redefines how industrial systems leverage large language models and autonomous agents. By introducing a tri-layered architecture integrating multi-agent collaboration (diagnostic/scheduling/digital twin agents), physics-informed knowledge engineering (RAG-augmented reasoning with DeepSeek), and self-optimizing LLMOps, we establish a new paradigm where (1) modular agent teams overcome single-model myopia through specialized role orchestration, (2) digital twins bridge physical causality and data-driven AI by generating simulated failure scenarios for LLM training, and (3) continuous refinement cycles enable systems to evolve via maintenance feedback—transforming static models into adaptive cognitive partners. This framework demonstrates that agentic AI transcends task automation; it embodies intentionality in industrial cognition, where LLMs become reasoning coordinators synthesizing sensor data, mechanical knowledge, and operational constraints into actionable foresight. Crucially, our tiered development approach (L1–L3) democratizes this intelligence, allowing low-code platforms like Dify to operationalize complex agent workflows while preserving enterprise-grade rigor through protocol-driven interoperability (A2A/MCP)—a critical enabler for scalable industrial AI agents.

This work remains a high-level architectural proposal, acting as a conceptual blueprint for a next-generation PdM system; more practical hurdles of implementation or evidence

that this intricate system would outperform simpler approaches need a deep dive. Current limitations include dependence on high-quality sensor data and historical fault records, challenges in cross-factory distributed collaboration under privacy constraints, and optimization hurdles for edge deployment in resource-constrained environments. Future research will address these gaps through synthetic data generation via physics-based models, federated learning architectures for secure multi-factory knowledge sharing, edge-optimized algorithms for low-resource scenarios, and standardized communication protocols ensuring robust multi-agent interaction. These advancements will further enhance the framework's scalability, adaptability, and security for broader industrial adoption.

Author Contributions: Conceptualization, F.H.; methodology, W.J. and F.H.; software, W.J.; validation, W.J.; investigation, W.J.; resources, F.H.; data curation, W.J.; writing—original draft preparation, W.J.; writing—review and editing, F.H. and W.J.; visualization, F.H. and W.J.; supervision, F.H.; project administration, F.H.; funding acquisition, F.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: All data generated or analyzed during this study are included in this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

A2A	Agent-to-Agent
AI	Artificial Intelligence
API	Application Programming Interface
AR	Augmented Reality
BaaS	Backend as a Service
CBM	Condition-Based Maintenance
DevOps	Development and Operations
DT	Digital Twin
FFT	Fast Fourier Transform
GPT	Generative Pre-trained Transformer
IIoT	Industrial Internet of Things
LLM	Large Language Model
LLMOps	Large Language Model Operations
MCP	Model Context Provider
MLOps	Machine Learning Operations
NLP	Natural Language Processing
OPC	Open Platform Communications
PdM	Predictive Maintenance
PLC	Programmable Logic Controller
PM	Preventive Maintenance
RAG	Retrieval Augmented Generation
RCM	Reliability-Centered Maintenance
ROI	Return on Investment
RUL	Remaining Useful Life
SCADA	Supervisory Control And Data Acquisition

References

1. Alves, F.; Badikyan, H.; Moreira, H.A.; Azevedo, J.; Moreira, P.M.; Romero, L.; Leitão, P. Deployment of a smart and predictive maintenance system in an industrial case study. In Proceedings of the 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE), Delft, The Netherlands, 17–19 June 2020; pp. 493–498. [CrossRef]
2. Moleda, M.; Momot, A.; Mrozek, D. Predictive maintenance of boiler feed water pumps using SCADA data. *Sensors* **2020**, *20*, 571. [CrossRef]
3. Salierno, G.; Morvillo, S.; Leonardi, L.; Cabri, G. An architecture for predictive maintenance of railway points based on big data analytics. In Proceedings of the International Conference on Advanced Information Systems Engineering, Grenoble, France, 8–12 June 2020; pp. 29–40.
4. Zhu, T.; Ran, Y.; Zhou, X.; Wen, Y. A survey on intelligent predictive maintenance (IPdM) in the era of fully connected intelligence. *IEEE Commun. Surv. Tutor.* **2025**. [CrossRef]
5. Stanton, I.; Munir, K.; Ikram, A.; El-Bakry, M. Predictive maintenance analytics and implementation for aircraft: Challenges and opportunities. *Syst. Eng.* **2022**, *26*, 216–237. [CrossRef]
6. Wang, J.; Liang, Y.; Zheng, Y.; Gao, R.X.; Zhang, F. An integrated fault diagnosis and prognosis approach for predictive maintenance of wind turbine bearing with limited samples. *Renew. Energy* **2020**, *145*, 642–650. [CrossRef]
7. Xiong, M.; Wang, H.; Fu, Q.; Xu, Y. Digital twin–driven aero-engine intelligent predictive maintenance. *Int. J. Adv. Manuf. Technol.* **2021**, *114*, 3751–3761. [CrossRef]
8. Wen, Y.; Rahman, M.F.; Xu, H.; Tseng, T.-L. Recent advances and trends of predictive maintenance from data-driven machine prognostics perspective. *Measurement* **2022**, *187*, 110276. [CrossRef]
9. Nguyen, K.T.; Medjaher, K. A new dynamic predictive maintenance framework using deep learning for failure prognostics. *Reliab. Eng. Syst. Saf.* **2019**, *188*, 251–262. [CrossRef]
10. Hafeez, T.; Xu, L.; Mcardle, G. Edge intelligence for data handling and predictive maintenance in IIOT. *IEEE Access* **2021**, *9*, 49355–49371. [CrossRef]
11. Killeen, P.; Ding, B.; Kiringa, I.; Yeap, T. IoT-based predictive maintenance for fleet management. *Procedia Comput. Sci.* **2019**, *151*, 607–613. [CrossRef]
12. Nunes, P.; Santos, J.; Rocha, E. Challenges in predictive maintenance—A review. *CIRP J. Manuf. Sci. Technol.* **2023**, *40*, 53–67. [CrossRef]
13. Dalzochio, J.; Kunst, R.; Pignaton, E.; Binotto, A.; Sanyal, S.; Favilla, J.; Barbosa, J. Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. *Comput. Ind.* **2020**, *123*, 103298. [CrossRef]
14. Cachada, A.; Barbosa, J.; Leitño, P.; Gcraldcs, C.A.; Deusdado, L.; Costa, J.; Teixeira, C.; Teixeira, J.; Moreira, A.H.; Moreira, P.M.; et al. Maintenance 4.0: Intelligent and predictive maintenance system architecture. In Proceedings of the 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Turin, Italy, 4–7 September 2018; pp. 139–146. [CrossRef]
15. Zonta, T.; Da Costa, C.A.; da Rosa Righi, R.; de Lima, M.J.; Da Trindade, E.S.; Li, G.P. Predictive maintenance in the Industry 4.0: A systematic literature review. *Comput. Ind. Eng.* **2020**, *150*, 106889. [CrossRef]
16. Andrianandrianina Johanesa, T.V.; Equeter, L.; Mahmoudi, S.A. Survey on AI applications for product quality control and predictive maintenance in industry 4.0. *Electronics* **2024**, *13*, 976. [CrossRef]
17. Jin, Q.; Chen, H.; Hu, F. Proposal of Industry 5.0-Enabled Sustainability of Product–Service Systems and Its Quantitative Multi-Criteria Decision-Making Method. *Processes* **2024**, *12*, 473. [CrossRef]
18. Abidi, M.H.; Mohammed, M.K.; Alkhalefah, H. Predictive maintenance planning for industry 4.0 using machine learning for sustainable manufacturing. *Sustainability* **2022**, *14*, 3387. [CrossRef]
19. Pothukuchi, S.N. LLMOps: A Comprehensive Guide to Deploying Large Language Models in Production. *IJSAT-Int. J. Sci. Technol.* **2025**, *16*, 1–12. [CrossRef]
20. Yang, L.; Luo, S.; Cheng, X.; Yu, L. Leveraging Large Language Models for Enhanced Digital Twin Modeling: Trends, Methods, and Challenges. *arXiv* **2025**, arXiv:2503.02167. [CrossRef]
21. Ferdousi, R.; Hossain, M.A.; Yang, C.; Saddik, A.E. Defecttwin: When llm meets digital twin for railway defect inspection. *arXiv* **2024**, arXiv:2409.06725. [CrossRef]
22. Russell-Gilbert, A. RAAD-LLM: Adaptive Anomaly Detection Using LLMs and RAG Integration. Ph.D. Thesis, Mississippi State University, Starkville, MS, USA, 2025.
23. Palma, G.; Cecchi, G.; Rizzo, A. Large Language Models for Predictive Maintenance in the Leather Tanning Industry: Multimodal Anomaly Detection in Compressors. *Electronics* **2025**, *14*, 2061. [CrossRef]
24. Timms, A.; Langbridge, A.; Antonopoulos, A.; Mygiakis, A.; Voulgari, E.; O'DOnncha, F. Agentic AI for Digital Twin. *Proc. AAAI Conf. Artif. Intell.* **2025**, *39*, 29703–29705. [CrossRef]
25. Xia, Y.; Xiao, Z.; Jazdi, N.; Weyrich, M. Generation of asset administration shell with large language model agents: Toward semantic interoperability in digital twins in the context of industry 4.0. *IEEE Access* **2024**, *12*, 84863–84877. [CrossRef]

26. Yoon, S.; Song, J.; Li, J. AI agent-based intelligent digital twins for building operations and maintenance. *J. Build. Eng.* **2025**, *108*, 130476. [CrossRef]
27. Lima Romero, M.; Suyama, R. Agentic AI for Intent-Based Industrial Automation. *arXiv* **2025**, arXiv:2506.04980. [CrossRef]
28. Tang, W.; Zhang, H.W.; Huang, J.; Wang, S.; Yu, F.; Yang, H.; Wang, Y. AgentBuilder: Automating agent creation via large language model-driven systems. *Neurocomputing* **2025**, *646*, 130476. [CrossRef]
29. Dong, L.; Lu, Q.; Zhu, L. AgentOps: Enabling Observability of LLM Agents. *arXiv* **2024**, arXiv:2411.05285. [CrossRef]
30. Hausi, A.M.; Litoiu, M.; Rivera, L.F.; Rasolroveicy, M.; Villegas, N.M.; Tamura, G.; Watts, I.; Erpenbach, E.; Shwartz, L. Proactive Continuous Operations using Large Language Models (LLMs) and AIOps. In Proceedings of the 33rd Annual International Conference on Computer Science and Software Engineering, Las Vegas, NV, USA, 11–14 September 2023; pp. 198–199.
31. Arai, K. Design of On-Premises Version of RAG with AI Agent for Framework Selection Together with Dify and DSL as Well as Ollama for LLM. *Int. J. Adv. Comput. Sci. Appl.* **2024**, *15*, 117–123. [CrossRef]
32. Pech, M.; Vrchota, J.; Bednář, J. Predictive maintenance and intelligent sensors in smart factory. *Sensors* **2021**, *21*, 1470. [CrossRef] [PubMed]
33. Samatas, G.G.; Moumgiakmas, S.S.; Papakostas, G.A. Predictive maintenance-bridging artificial intelligence and iot. In Proceedings of the 2021 IEEE World AI IoT Congress (AIIoT), Seattle, WA, USA, 10–13 May 2021; pp. 413–419. [CrossRef]
34. Chen, S.; Bekar, E.T.; Bokrantz, J.; Skoogh, A. AI-enhanced digital twins in maintenance: Systematic review, industrial challenges, and bridging research–practice gaps. *J. Manuf. Syst.* **2025**, *82*, 678–699. [CrossRef]
35. Aivaliotis, P.; Georgoulias, K.; Chryssolouris, G. The use of Digital Twin for predictive maintenance in manufacturing. *Int. J. Comput. Integr. Manuf.* **2019**, *32*, 1067–1080. [CrossRef]
36. Hosamo, H.H.; Svennevig, P.R.; Svidt, K.; Han, D.; Nielsen, H.K. A Digital Twin predictive maintenance framework of air handling units based on automatic fault detection and diagnostics. *Energy Build.* **2022**, *261*, 111988. [CrossRef]
37. Yang, H.; Siew, M.; Joe-Wong, C. An llm-based digital twin for optimizing human-in-the loop systems. In Proceedings of the 2024 IEEE International Workshop on Foundation Models for Cyber-Physical Systems & Internet of Things (FMSys), Hong Kong, China, 13–15 May 2024; pp. 26–31. [CrossRef]
38. Cao, Q.; Zanni-Merk, C.; Samet, A.; Reich, C.; De Beuvron, F.D.; Beckmann, A.; Giannetti, C. KSPMI: A knowledge-based system for predictive maintenance in industry 4.0. *Robot. Comput. Integr. Manuf.* **2022**, *74*, 102281. [CrossRef]
39. Russell-Gilbert, A.; Sommers, A.; Thompson, A.; Cummins, L.; Mittal, S.; Rahimi, S.; Seale, M.; Jaboure, J.; Arnold, T.; Church, J. Aad-llm: Adaptive anomaly detection using large language models. In Proceedings of the 2024 IEEE International Conference on Big Data (BigData), Washington, DC, USA, 15–18 December 2024; pp. 4194–4203. [CrossRef]
40. Ayvaz, S.; Alpay, K. Predictive maintenance system for production lines in manufacturing: A machine learning approach using IoT data in real-time. *Expert Syst. Appl.* **2021**, *173*, 114598. [CrossRef]
41. Zhang, N.; Vergara-Marcillo, C.; Diamantopoulos, G.; Shen, J.; Tziritas, N.; Bahsoon, R.; Theodoropoulos, G. Large language models for explainable decisions in dynamic digital twins. *arXiv* **2024**, arXiv:2405.14411. [CrossRef]
42. Han, X.; Wang, Z.; Xie, M.; He, Y.; Li, Y.; Wang, W. Remaining useful life prediction and predictive maintenance strategies for multi-state manufacturing systems considering functional dependence. *Reliab. Eng. Syst. Saf.* **2021**, *210*, 107560. [CrossRef]
43. Calabrese, M.; Cimmino, M.; Fiume, F.; Manfrin, M.; Romeo, L.; Ceccacci, S.; Paolanti, M.; Toscano, G.; Ciandrini, G.; Carrotta, A.; et al. SOPHIA: An event-based IoT and machine learning architecture for predictive maintenance in industry 4.0. *Information* **2020**, *11*, 202. [CrossRef]
44. Yang, H.; LaBella, A.; Desell, T. Predictive maintenance for general aviation using convolutional transformers. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 12636–12642. [CrossRef]
45. Aivaliotis, P.; Georgoulias, K.; Arkouli, Z.; Makris, S. Methodology for enabling digital twin using advanced physics-based modelling in predictive maintenance. *Procedia CIRP* **2019**, *81*, 417–422. [CrossRef]