Journal of *Marine Science and Engineering*

MDPI

Article

AISStream-MCP: A Real-Time Memory-Augmented Question-Answering System for Maritime Operations

Sien Chen [1,2], Ruoxian Zhao [3], Jian-Bo Yang [3] and Yinghua Huang [4,*]

[1] School of Navigation, Jimei University, Xiamen 361021, China; sienchen@jmu.edu.cn
[2] School of Management, Xiamen University, Xiamen 361005, China; sienchenxm@xmu.edu.cn
[3] Alliance Manchester Business School, The University of Manchester, Manchester M15 6PB, UK; gisellezhao@outlook.com (R.Z.); jian-bo.yang@manchester.ac.uk (J.-B.Y.)
[4] Lucas College and Graduate School of Business, San Jóse State University, San Jóse, CA 95192, USA
* Correspondence: yinghua.huang@sjsu.edu

Abstract

Ports and maritime operations generate massive real-time data streams, particularly from Automatic Identification System (AIS) signals, which are challenging to query effectively using natural language. This study proposes a prototype AISStream-MCP, a memory-augmented real-time maritime question-answering (QA) system that integrates live AIS data streaming with a Model Context Protocol (MCP) toolchain to support port operations decision-making. The system combines a large language model (LLM) with four MCP-enabled modules: persistent dialogue memory, live AIS data query, knowledge graph lookup, and result evaluation. We hypothesize that augmenting an LLM with domain-specific tools significantly improves QA performance compared to systems without memory or live data access. To test this hypothesis, we developed two prototype systems (with and without MCP framework) and evaluated them on 30 queries across three task categories: ETA prediction, anomaly detection, and multi-turn route queries. Experimental results demonstrate that AISStream-MCP achieves 88% answer accuracy (vs. 75% baseline), 85% multi-turn coherence (vs. 60%), and 38.7% faster response times (4.6 s vs. 7.5 s), with user satisfaction scores of 4.6/5 (vs. 3.5/5). The improvements are statistically significant ($p < 0.01$), confirming that memory augmentation and real-time tool integration effectively enhance maritime QA capabilities. Specifically, AISStream-MCP improved ETA prediction accuracy from 80% to 90%, anomaly detection from 70% to 85%, and multi-turn query accuracy from 65% to 88%. This approach shows significant potential for improving maritime situational awareness and operational efficiency.

Keywords: maritime question-answering; model context protocol; real-time AIS data; knowledge augmentation; port decision support

Academic Editor: Marco Cococcioni

Received: 9 August 2025
Revised: 3 September 2025
Accepted: 5 September 2025
Published: 11 September 2025

Citation: Chen, S.; Zhao, R.; Yang, J.-B.; Huang, Y. AISStream-MCP: A Real-Time Memory-Augmented Question-Answering System for Maritime Operations. *J. Mar. Sci. Eng.* **2025**, *13*, 1754. https://doi.org/10.3390/jmse13091754

1. Introduction

Modern ports and maritime transport networks produce vast amounts of real-time data from vessels and infrastructure, yet it remains difficult for operators to query this information directly and obtain timely insights for decision-making. For example, port controllers may need to know "When will ship X arrive given its current course and speed?" or "Has vessel Y deviated from its planned route right now?". Answering such questions is challenging because it requires accessing up-to-the-minute data and contextual knowledge. Conventional maritime information systems and databases do not readily support natural language queries, and human operators often rely on manual data lookup and experience,

which can be inefficient and error-prone. These gaps motivate research into intelligent question-answering (QA) assistants that can leverage streaming maritime data to provide on-demand decision support. For example, deep-learning-based ETA prediction models have been employed in port scheduling and berth allocation [1]. Additionally, sequence-modeling methods using Automatic Identification System (AIS) data have been applied for ETA forecasting in container or bulk ports [2]. These studies highlight the feasibility of real-time data fusion and ML techniques in maritime decision support.

Recent advances in artificial intelligence have opened new possibilities for such systems. On one hand, structured knowledge bases and knowledge graphs (KGs) have been developed for the maritime domain—for example, ontologies modeling vessel behavior and maritime rules [3] and knowledge graphs of vessel incidents [4]—which provide rich factual data. On the other hand, large language models (LLMs) such as GPT-3 and GPT-4 have demonstrated impressive capabilities in understanding and generating natural language [5,6]. By combining KGs with LLMs, a QA model's reasoning can be grounded in real and up-to-date data, thereby reducing hallucination errors and improving domain-specific accuracy. Initial studies in complex domains like finance, medicine, and maritime operations have explored this approach, indicating that augmenting LLMs with relevant knowledge can significantly improve answer correctness. Recent advances in large language model-based agents have shown remarkable capabilities in complex reasoning and tool use [7,8]. Augmented language models that combine retrieval mechanisms with generation have demonstrated superior performance in knowledge-intensive tasks [9,10].

However, existing maritime QA solutions remain limited in two key aspects: context memory and real-time data integration. Most prior systems—such as simple knowledge-based query interfaces or static rule-based assistants—handle one question at a time without remembering dialogue context, forcing users to repeat vessel identifiers or details in each query. Moreover, these systems often rely on static or manually updated data; for instance, a recent KG-augmented QA prototype for vessel identification improved accuracy using historical AIS facts, but it could not ingest live streaming data or adapt to unfolding events. In practice, this means questions like "Has vessel X deviated from its route right now?" or "Given its current speed, when will ship Y arrive at Port Z?" cannot be answered accurately by traditional systems, since they lack real-time situational awareness. Additionally, without dialogue memory, if a user asks a follow-up (e.g., "Where is it heading after that?"), a standard QA system loses the context, leading to confusion or incorrect answers.

To address these gaps, we propose AISStream-MCP, a prototype intelligent maritime QA architecture that integrates real-time AIS data streams with an LLM through the Model Context Protocol (MCP) framework. The core idea is to equip the LLM-based assistant with tools for remembering conversation context and fetching live domain data during its reasoning process. We expect that this memory-augmented, real-time approach will significantly outperform a conventional QA system without such enhancements. Specifically, we hypothesize that incorporating a persistent memory and live data access will lead to more accurate and coherent answers in maritime QA tasks. We design an experimental study to test this hypothesis by comparing AISStream-MCP against a baseline system lacking memory and live data. In summary, our contributions are as follows:

(1) We design a new memory-augmented, real-time QA architecture for the maritime domain. To our knowledge, this is the first system architecture to integrate an open MCP-based toolchain (memory, live data, and knowledge graph query) with an LLM to support port operations.
(2) We develop a working prototype (AISStream-MCP) and a comparable non-MCP baseline and perform extensive experiments on three representative maritime QA tasks. The tasks include estimated time of arrival (ETA) prediction, anomaly detection (route

deviation), and multi-turn route-related queries. We also introduce an interactive web-based evaluation platform to log tool usage and user interactions during testing.

(3) Through quantitative metrics and user evaluations, we demonstrate that the MCP-enhanced system significantly outperforms the baseline in answer accuracy, dialogue continuity, and responsiveness. We provide a detailed analysis of results, showing improved answer correctness and multi-turn coherence, faster average response time, and higher user satisfaction for AISStream-MCP. The improvements are statistically significant, confirming the effectiveness of memory and tool integration for maritime QA.

(4) We outline future extensions of the system, including incorporation of multi-modal data and multi-language support. This paves the way for next-generation intelligent maritime assistants that can integrate visual sensor data (e.g., weather, radar, remote sensing imagery) and handle queries in various languages, thereby greatly expanding the system's applicability. We also discuss considerations for real-world deployment, such as secure MCP server integration in port IT environments, handling high query volumes, and ensuring data privacy.

Real-time and accurate maritime information access is critical for supporting behavior adaptation, port coordination, and navigation safety, which are central to modern intelligent transportation systems. The proposed system not only enhances technical responsiveness but also holds potential to influence operational decisions in maritime domains.

In contrast to ad-hoc API integration, the MCP framework offers several distinctive advantages: it provides a unified and standardized protocol for connecting heterogeneous tools, supports modular extensibility for integrating new services, includes built-in capabilities for persistent memory servers, and enables secure deployment within containerized environments. Collectively, these structural features make MCP especially well-suited for real-time maritime QA applications, where robustness and scalability are essential.

2. Literature Review

2.1. Maritime Information Systems and Decision Support

The maritime domain has seen growing efforts in constructing knowledge bases and ontologies to support intelligent analytics and QA. For example, Zhong & Wen [3] modelled ship behaviors and navigational rules (COLREGs) in an ontological framework to enable rule-based reasoning about vessel interactions. Shiri et al. [11] proposed a semi-automated method to construct probabilistic maritime knowledge graphs for anomaly detection and risk analysis, reflecting the need to capture uncertain relationships in maritime data. In a related vein, Liu & Cheng [4] developed a Maritime Accident Knowledge Graph (MAKG) focused on incidents and accidents, to aid in accident analysis and management. These knowledge repositories lay a foundation for maritime QA systems by organizing factual domain information. However, querying such knowledge bases typically requires specialized query languages or custom interfaces, limiting their accessibility to end-users. Our work aims to leverage these rich maritime knowledge sources by using an LLM as a natural language interface, which makes querying more intuitive and convenient.

2.2. Knowledge-Augmented and Memory-Based QA in Transportation Systems

Pre-trained LLMs have achieved remarkable success in general QA tasks, but their performance in specialized domains can degrade without domain-specific augmentation. Hu et al. [12] found that off-the-shelf language models perform poorly on knowledge graph-based QA in specific domains unless they are enhanced with external information or tools. This finding underscores that retrieval-augmented generation and tool use are critical for applying LLMs in specialized fields. This finding underscores that retrieval-

augmented generation and tool use are critical for applying LLMs in specialized fields. Domain-specific language models [13,14] and tool-augmented reasoning frameworks [15] have emerged as promising solutions for addressing these limitations. In recent years, a number of frameworks have been proposed to integrate LLMs with external knowledge and data sources. Some approaches focus on knowledge graph retrieval and prompting (e.g., [16–18]), demonstrating that injecting relevant facts from KGs into LLM prompts can substantially improve accuracy in complex question answering. Others explore retrieval-augmented generation and tool use for factual correctness (e.g., [19–21]), where the LLM is empowered to call APIs, databases, or calculators during its reasoning. This trend aligns with the open-source AI community's development of standardized tool-use protocols. Notably, Anthropic [22] introduced the Model Context Protocol (MCP), which provides a unified open interface for connecting LLMs with various services, databases, and APIs. MCP replaces ad-hoc integrations with a consistent protocol, enabling AI agents to seamlessly query multiple data sources. Building on this, the concept of an MCP-based memory server has been proposed to allow an AI assistant to share persistent conversational context across sessions. These advances suggest that an LLM equipped with tools for retrieval and memory can overcome many limitations of standalone LLMs in domain-specific tasks. In the maritime context, connecting an LLM to live port databases, meteorological information, and streaming AIS data could greatly enhance its practical usefulness.

2.3. Real-Time Data Integration and Application in Transport Intelligence

Traditional QA systems struggle with real-time reasoning due to their dependence on static corpora. In maritime contexts, integrating real-time AIS streams is essential for situational awareness. AISStream.io enables global vessel monitoring via WebSocket, and LLMs can retrieve up-to-date vessel states using tools like ais.stream (AISStream.io API, version 1.3) through MCP. Furthermore, memory systems support long-context retention across multi-turn dialogues, enhancing coherence. These integrations—memory, real-time data, and domain knowledge—form the foundation of dynamic maritime QA assistants like AISStream-MCP. Our research follows this direction: we integrate multiple tools (memory, live data query, graph database access, etc.) with an LLM to create a QA system that can reliably handle complex port operation queries with real-time awareness.

3. Methodology

3.1. System Architecture

The proposed AISStream-MCP framework can be conceptualized through the lens of an autonomous agent operating within a partially observable environment [23]. In this paradigm, the LLM acts as the agent's reasoning core, or "brain". The environment consists of the user's intent and the vast, dynamic maritime information space. The agent's "observations" are not just the user's query but also the real-time data and static facts it acquires through its "sensors"—the MCP tools. The system's objective is to execute a policy, formulated as a sequence of tool invocations and internal reasoning steps, to reach a goal state: providing a correct, coherent, and context-aware answer. This agentic perspective elevates the system from a simple input-output model to a proactive, knowledge-seeking entity, which is central to our methodology.

Our proposed system, AISStream-MCP, is built on an architecture that tightly integrates an LLM with four domain-specific tool modules via the Model Context Protocol (MCP). Figure 1 provides an overview of the architecture. At the core is the LLM-based QA Engine, which interacts with users in natural language. Surrounding it are four tools: a Persistent Memory module, a Live AIS Data Stream interface, a Port Knowledge Graph database interface, and a Result Evaluation module. The LLM orchestrates these compo-

nents through MCP's unified interface. MCP provides a standardized way for the LLM (acting as an AI agent) to invoke external services and data sources [22]. In our implementation, an MCP-compatible server hosts the tool functions, and the LLM issues tool requests (e.g., data queries, memory read/write operations) as text commands, which the MCP server executes. This design replaces ad-hoc integration of tools with a consistent protocol, allowing the QA system to seamlessly combine multiple data sources and functionalities during its reasoning process. By leveraging the open MCP standard, AISStream-MCP effectively bridges the gap between dynamic maritime data and the LLM's natural language reasoning capabilities.

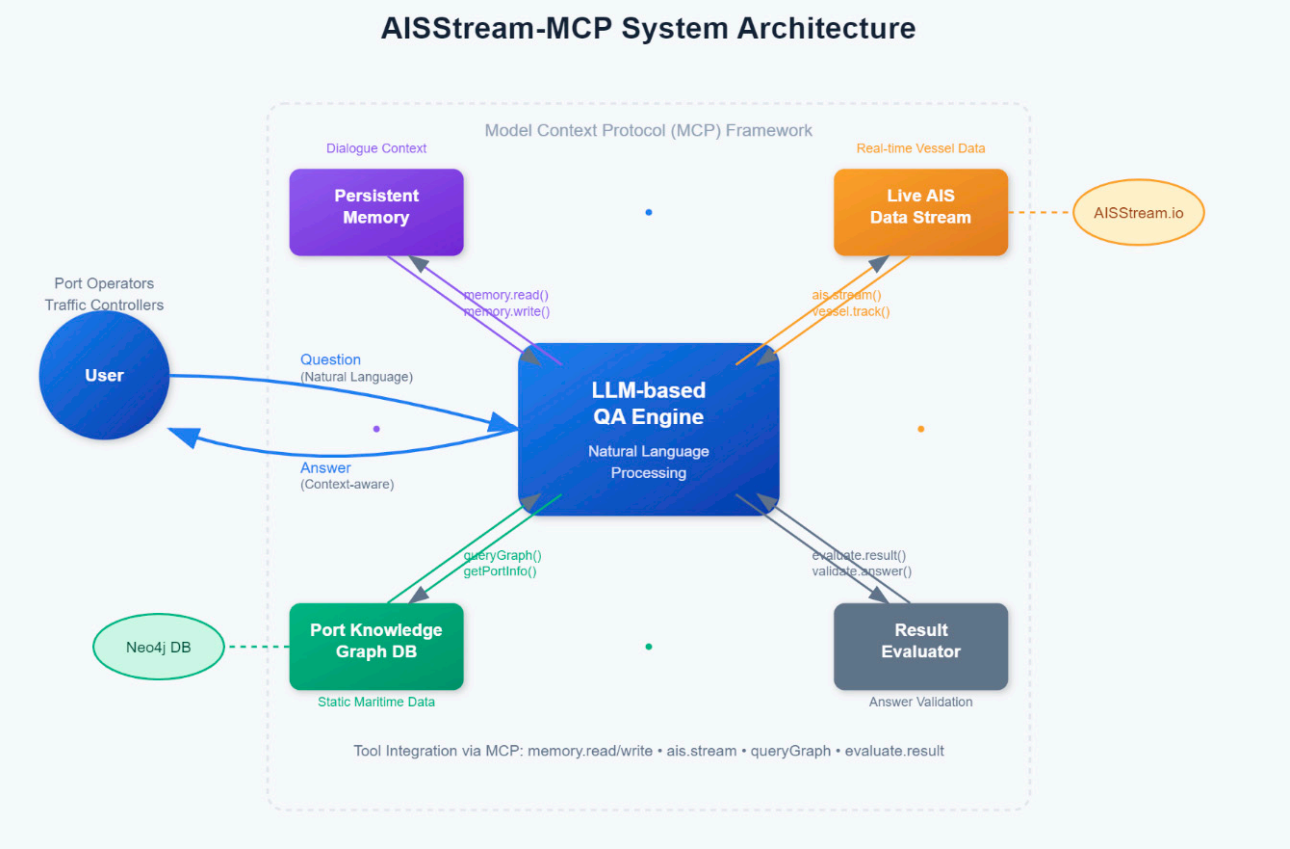


Figure 1. AISStream-MCP system architecture. The LLM-based QA engine interacts with four integrated tools via MCP: a persistent memory for dialogue context, live AIS data stream access, a port knowledge graph database, and a result evaluation module.

In our reference deployment, the MCP layer introduced a communication latency of consistently less than 100 ms, which is negligible relative to the inference time of the LLM. To ensure system security, the prototype employs three key mechanisms: encrypted WebSocket connections for all tool communications, stringent API key management for service authentication, and container-based deployment with strict access control lists (ACLs) to isolate services.

We formally define the system's response behavior as a function:

$$A_{LLM}(q_t) = f(q_t, C_t, M_t, D_t) \tag{1}$$

where q_t is the current user query, C_t is the dialogue context, M_t denotes persistent memory, and D_t represents real-time data such as AIS feeds. This function models how the LLM coordinates tools and memory to generate responses. In this architecture, Persistent Memory plays a crucial role in maintaining context over a dialogue. The memory module stores

key information from user queries and the assistant's answers, allowing the LLM to recall prior context in subsequent turns. The Live AIS Data Stream tool (Figure 2) provides the LLM with real-time vessel position and movement data. Our system subscribes to a global AIS feed (via AISStream.io) to receive up-to-the-minute AIS messages. The Port Knowledge Graph (KG) is a structured database of static maritime information. We integrate a Neo4j-based port knowledge graph and expose it via a queryGraph tool (Neo4j version 4.4.12), leveraging recent advances in schema-aware text-to-graph conversion [24,25] and unified approaches to LLM-knowledge graph integration [26]. Finally, the Result Evaluation module acts as a post-processor and verifier for the LLM's generated answers. As shown in Figure 2, the real-time AIS Stream WebSocket connection is implemented to enable vessel tracking.

ais-stream-mcp-server.js

```
// WebSocket-based AIS Stream Connection with real-time vessel tracking
async connectAISStream( args ) {
   const { apiKey, boundingBox, connectionId= 'maritime-emergency'} =args;
   const ws = new WebSocket( 'wss://stream.aisstream.io/v0/stream'

   return new Promise((resolve, reject) => {
      ws.on( 'open', () => {
         // Send subscription message with geographical boundaries
         const subscription = {
            APIKey: apiKey,
            BoundingBoxes: [[[
               boundingBox.topLeft[0],boundingBox.topLeft[1],
               boundingBox.bottomRight[0],boundingBox.bottomRight[1]]],
            FilterMessageTypes: this.convertMessageTypesToNames(messageTypes)
         };
         ws.send( JSON.stringify( subscription));
         console.log( `[WebSocket] Connection ${connectionId} established`
         resolve({success: true, connectionId status: 'connected'});
      });

      ws.on( 'message', (data) => {
         try {
            const aisMessage= JSON.parse (data.toString());
            if ( aisMessage.MetaData) {
               // Process vessel position and update real-time tracking
               console.log (`[AIS] Vessel ${aisMessage.MetaData.MMSI} at `
                  `${aisMessage.Message.PositionReport.Latitude}, `
                  `${aisMessage.Message.PositionReport.Longitude}`
               this.handleAISMessage( connectionId aisMessage);

               // Update vessel positions for real-time tracking
               const mmsi = aisMessage.MetaData.MMSI;
               const pos = aisMessage.Message.PositionReport;
               this.vessels.set ( mmsi, {
                  mmsi, lat: pos.Latitude, lon : pos.Longitude,
                  lastUpdate: new Date ().toISOString()
               });
            }
         } catch ( e ) {
            console.error ( "[WebSocket] Message parsing error);"
         }
      });
   });
}
```

Figure 2. Real-time AIS stream WebSocket connection implementation. This implementation establishes a WebSocket connection to AISStream.io for real-time vessel tracking.

3.2. MCP Tool Integration and Workflow

When a user poses a question to AISStream-MCP, the system processes it in the following sequence. First, the user query in natural language is received by the LLM QA Engine. The LLM, guided by its prompt and system instructions, analyzes the query to determine what external information or functions are needed. It can then issue MCP tool commands embedded in a special format within its generated "thought" sequence.

Once the query is processed, the system infers the task type and corresponding command as:

$$\tau = \arg \max_{\tau'} P(\tau' \mid q_t, C_t), c_t = \pi(\tau, D_t, M_t) \quad (2)$$

where $P(\cdot)$ is the task classification model and $\pi(\cdot)$ maps task types to executable MCP commands. For instance, for a question like "Has vessel Alpha deviated from its route?", the LLM might decide to call the ais.stream tool to obtain Alpha's latest coordinates and compare with the planned route. It formulates a command such as <call> ais.stream("Alpha"), which the MCP server executes, returning live AIS data. The LLM receives these data and incorporates them into its reasoning. The entire workflow happens in real time, typically within a few seconds, enabling an interactive QA experience. This approach follows established patterns in retrieval-augmented generation [27,28] and tool-augmented language models [29]. The final answer is synthesized through a tool-aware fusion function:

$$y_t = \phi\left(\text{LLM}_{gen}(q_t), R_t^{MCP}, R_t^{AIS}, M_t\right) \tag{3}$$

where $\phi(\cdot)$ integrates LLM's initial output with results from tool calls and memory for coherence and factual grounding.

As shown in Figure 3, the MCP command parser and router implementation serves as the central command router within this workflow. The sequence of tool invocation across multiple turns is illustrated in Figure 4, which traces the complete interaction flow for a two-part query using memory, real-time data, and the knowledge graph.

mcp_interface.py

```python
class MCPInterface:
    def parse_and_execute(self, cmd: str, session_id: str):
        """
        Parse <call>module.function(params)> commands and route to appropriate modules.
        Implements the tool orchestration layer shown in Figure 1.
        """
        # Parse MCP command format using regex
        m = re.match(r"<call>\s*([a-zA-Z_][a-zA-Z0-9_]*)\.([a-zA-Z_][a-zA-Z0-9_]*)\((.*?)\)\s*>", cmd.strip())

        if m:
            module, function, raw_params = m.group(1), m.group(2), m.group(3)
            params = self._parse_params(raw_params)

            # Route to appropriate module based on parsed command
            if module == "memory":
                if function == "read":
                    result = self.memory.memory_read(session_id)
                elif function == "write":
                    result = self.memory.memory_write(session_id, params[0])

            elif module == "ais":
                if function == "stream":
                    result = self.call_ais_mcp_tool("ais.stream", {"target": params[0]})

            elif module == "queryGraph":
                result = self.kg.query(params[0] if params else "")

            elif module == "evaluate":
                result = self.evaluator.evaluate(params[0] if params else "")

            return {
                "module": module,
                "function": function,
                "params": params,
                "result": result
            }
```

Figure 3. MCP Command Parser and Router implementation. The MCP interface class serves as the central command router. Note: "#" indicates code comments, and "*" follows standard Python syntax (e.g., argument unpacking).

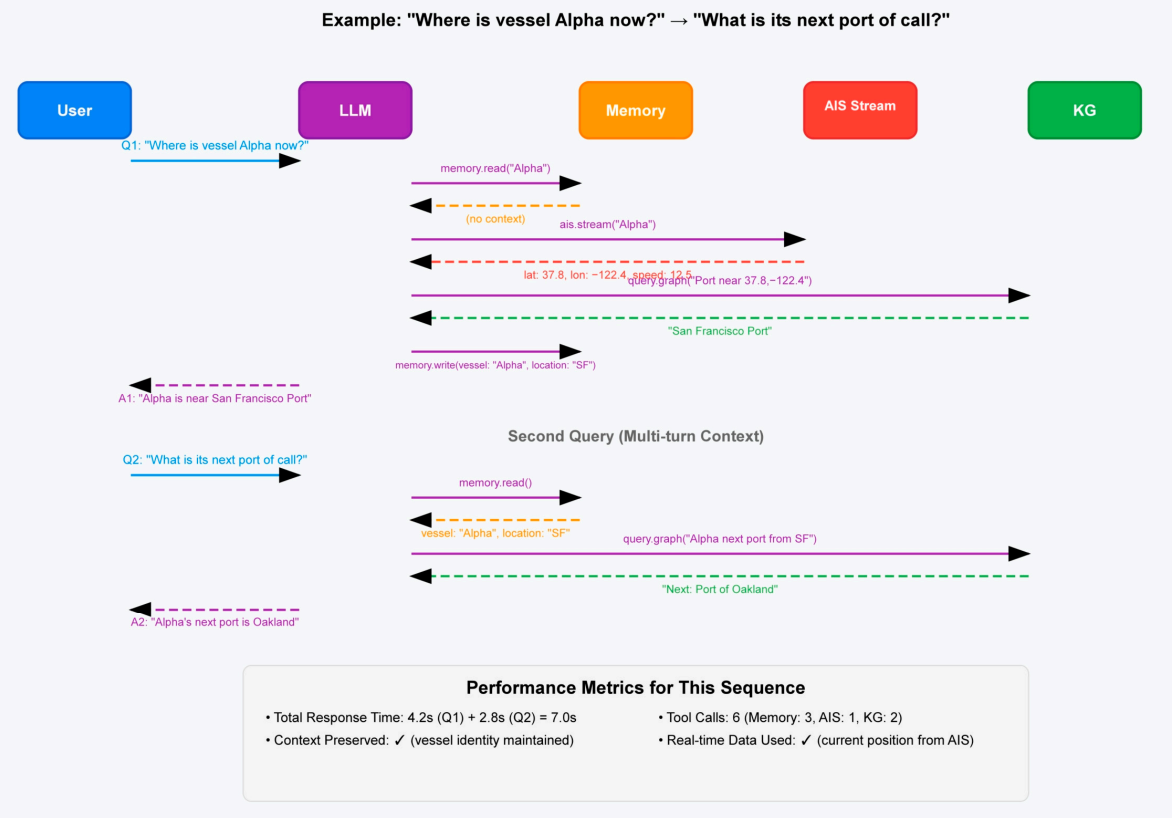


Figure 4. Tool invocation sequence for multi-turn query processing. The diagram traces the complete interaction flow for a two-part query, showing how the system utilizes memory, real-time data, and the knowledge graph.

4. Case Study and Experiment Design

4.1. Experimental Environment

The experiments were conducted on a server with an Intel Xeon Gold 6248R CPU, 128 GB RAM, and an NVIDIA Tesla V100 GPU. The software environment is using Ubuntu 20.04 LTS, Python 3.8.10, Neo4j 4.4.12, Docker 20.10.21, and GPT-3.5-turbo.

4.2. Baseline System Implementation

To quantify the performance improvements, our experimental design employs an "enhanced system (MCP toolchain)" versus "baseline system (Neo4j knowledge graph)" comparison. The Neo4j baseline represents the current mainstream approach of combining a structured knowledge base with an LLM. To ensure fair comparison, all experimental queries, data access points, and interfaces remain consistent between both systems, differing only in their capabilities.

While a baseline with a frequently synchronized database (e.g., batch updates every minute) could be considered, we argue that such a design fundamentally fails to capture the essence of "real time" required for critical maritime operations. Scenarios like immediate collision risk assessment or responding to sudden vessel deviations demand sub-second data latency, which can only be achieved through a direct, streaming connection like WebSocket. A batch-updated system, by its nature, introduces a latency floor equivalent to its update interval, rendering it inadequate for these high-stakes use cases. Therefore, our chosen baseline, representing a static-knowledge paradigm, serves to create the clearest possible contrast against the true real-time, streaming paradigm that our work proposes, thereby isolating the core scientific question of our research. As shown in Figure 5, the Neo4j baseline implementation illustrates the static-knowledge paradigm used for comparison.

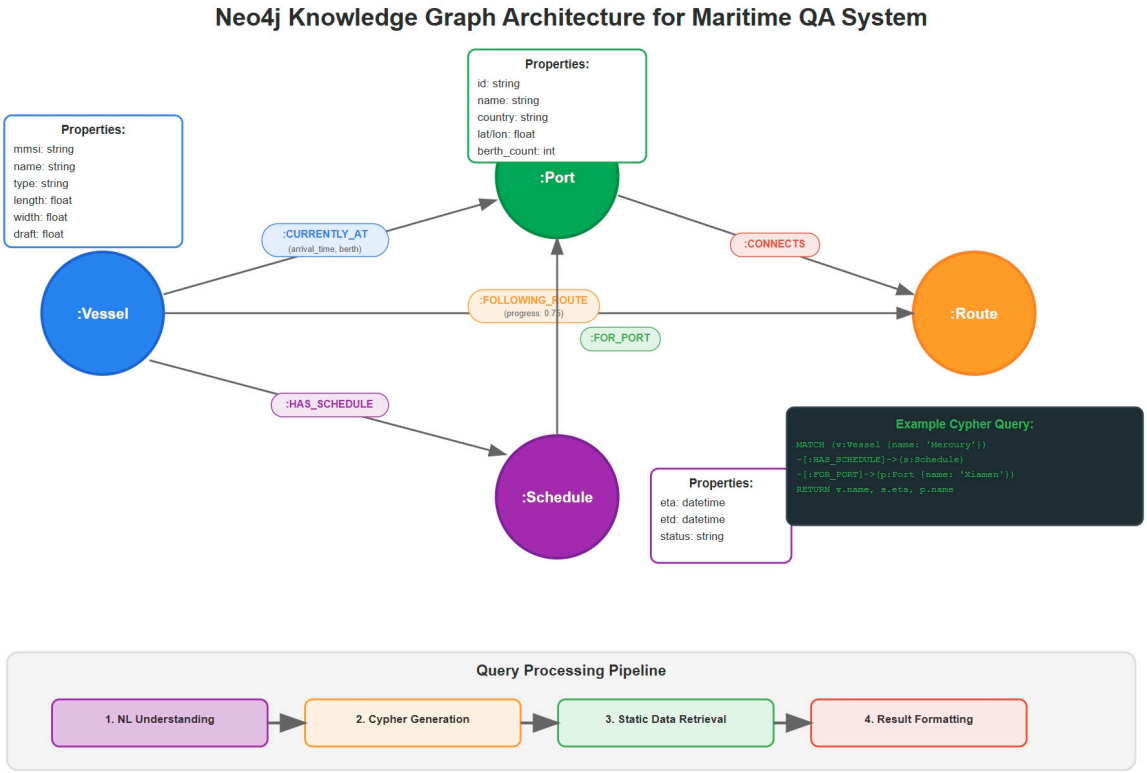


Figure 5. Neo4j baseline implementation details.

4.3. Test Query Specifications

We conducted a comparative study between our proposed system and the baseline system without MCP enhancements, designed around a port operations scenario. Figure 6 illustrates the complete experimental process. Three representative QA task categories were used, reflecting common information needs [30,31]: ETA Prediction, Anomaly Detection, and Multi-turn Route Queries. To quantify multi-turn coherence, we define the following metric:

$$CR = \frac{1}{N}\sum_{i=1}^{N} 1[y_i \text{ respects } C_{<i}] \tag{4}$$

which measures how well the system preserves prior context. In total, 30 queries were prepared. Each query came with an expected correct answer (ground truth) obtained from historical data or domain experts. To mitigate selection bias, the query scenarios were designed in collaboration with the industry practitioners on our evaluation panel before the final system was implemented. The scenarios were derived from a random sampling of actual operational logs from the past 6 months, ensuring they reflect a realistic distribution of common and critical information needs, rather than being crafted to favor our system's capabilities.

The evaluation was conducted with a panel of five participants (N = 5). The panel consisted of two senior researchers specializing in maritime informatics and three industry practitioners with over 10 years of experience in port logistics and vessel traffic services. This composition ensures that the evaluation captures both academic rigor and practical operational relevance. Each participant evaluated the full set of 30 queries, with the system order randomized for each user to mitigate learning effects. We used a five-point Likert scale for user satisfaction.

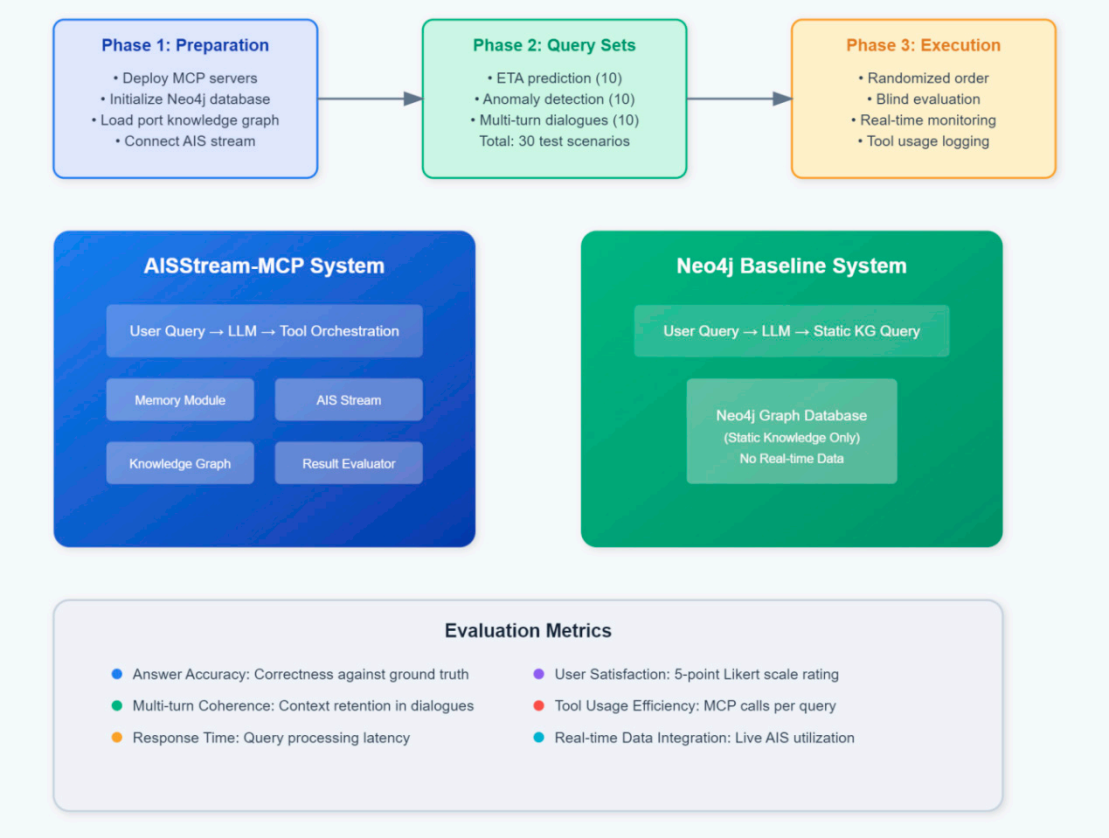


Figure 6. Experimental workflow diagram. The experiment consists of preparation, query definition, and a blind execution phase.

The 30 test queries were carefully designed to cover realistic port operation scenarios. Table 1 provides the complete query set as follows.

Table 1. Complete Test Query Specifications.

ID	Category	Query	Ground Truth Source	Key Evaluation Aspect
		ETA Prediction Queries		
ETA-01	ETA	When will vessel Mercury arrive at Port of Xiamen?	AIS position + speed calculation	Real-time adjustment
ETA-02	ETA	What is the expected arrival time for Neptune considering current weather?	AIS + weather API	Multi-source fusion
ETA-03	ETA	Calculate arrival time for vessel Jupiter at Berth 5	AIS + port scheduling	Berth-specific ETA
ETA-04	ETA	Is vessel Saturn running on schedule?	Schedule vs. actual position	Delay detection
ETA-05	ETA	When will the container ship Venus reach the pilot station?	AIS + port geography	Waypoint calculation
ETA-06	ETA	Estimate arrival for bulk carrier Mars with current speed	Real-time speed data	Speed-based prediction
ETA-07	ETA	Will vessel Uranus arrive before high tide?	AIS + tidal data	Time constraint check
ETA-08	ETA	Update ETA for delayed vessel Pluto	Historical + real-time	Dynamic updating
ETA-09	ETA	Calculate new arrival time after route change	Route modification	Recalculation ability
ETA-10	ETA	Batch ETA query for all inbound vessels	Multiple vessel tracking	Scalability test

Table 1. *Cont.*

ID	Category	Query	Ground Truth Source	Key Evaluation Aspect
		Anomaly Detection Queries		
AD-01	Anomaly	Has vessel Alpha deviated from its planned route?	Planned vs. actual route	Deviation detection
AD-02	Anomaly	Detect unusual speed patterns for vessel Beta	Speed history analysis	Behavioral anomaly
AD-03	Anomaly	Is vessel Gamma anchored in an unusual location?	Anchor zone validation	Position anomaly
AD-04	Anomaly	Alert if any vessel enters restricted area	Geofence monitoring	Zone violation
AD-05	Anomaly	Has vessel Delta been stationary too long?	Movement patterns	Idle detection
AD-06	Anomaly	Identify vessels not broadcasting AIS	Signal continuity	Communication anomaly
AD-07	Anomaly	Detect collision risk between vessels	CPA/TCPA calculation	Safety monitoring
AD-08	Anomaly	Find vessels with suspicious behavior patterns	Multi-factor analysis	Complex anomaly
AD-09	Anomaly	Check if vessel Epsilon changed destination	Destination tracking	Plan modification
AD-10	Anomaly	Monitor compliance with speed restrictions	Speed limit zones	Regulation compliance
		Multi-turn Route Queries		
MT-01	Multi-turn	Q1: Where is vessel Zeta now? Q2: What's its next port?	Context preservation	Basic context
MT-02	Multi-turn	Q1: Track vessel Eta movement Q2: How long at current location?	Temporal context	Time tracking
MT-03	Multi-turn	Q1: Show vessels from Singapore Q2: Which arrives first?	Set context	Comparison context
MT-04	Multi-turn	Q1: Status of tanker Theta Q2: Is it fully loaded? Q3: ETA?	Multiple attributes	Extended context
MT-05	Multi-turn	Q1: Find vessel MMSI 123,456,789 Q2: Its cargo type? Q3: Destination?	ID resolution	Reference tracking
MT-06	Multi-turn	Q1: List container ships Q2: Filter by size Q3: Show routes	Progressive filtering	Query refinement
MT-07	Multi-turn	Q1: Vessel Iota position Q2: Distance to port Q3: Fuel sufficiency?	Calculation chain	Dependent queries
MT-08	Multi-turn	Q1: Port congestion status Q2: Affected vessels Q3: Suggest alternatives	Problem solving	Complex reasoning
MT-09	Multi-turn	Q1: Weather at vessel Kappa location Q2: Impact on schedule Q3: Notify if delayed	Condition monitoring	Proactive alerts
MT-10	Multi-turn	Q1: Historical routes of Lambda Q2: Most frequent port Q3: Average duration	Historical analysis	Pattern recognition

4.4. Statistical Analysis Methods

The following significance Testing methods were used.

- Answer Accuracy Comparison: A two-proportion z-test was used.

$$z = \frac{\hat{p}\text{MCP} - \hat{p}\text{Baseline}}{\sqrt{\hat{p}(1-\hat{p})\left(\frac{1}{n_{\text{MCP}}} + \frac{1}{n_{\text{Baseline}}}\right)}} \tag{5}$$

- Response Time Analysis: A paired t-test was applied.

$$t = \frac{\overline{d}}{s_d/\sqrt{n}}, \text{Cohen's } d = \frac{\overline{d}}{s_d} \tag{6}$$

- Multi-turn Coherence: McNemar's test was used.

$$\chi^2 = \frac{(b-c)^2}{b+c} \tag{7}$$

- User Satisfaction Ratings: Compared using Wilcoxon signed-rank test.
- Multiple Comparison Correction: Bonferroni correction was applied:

$$\alpha_{\text{adjusted}} = \frac{0.05}{4} = 0.0125 \tag{8}$$

4.5. Ablation, Sensitivity, and Load Test Protocols

Ablation Study: We performed a targeted ablation on 15 representative queries, evaluating five configurations: Full (memory + AIS + KG), –Memory, –AIS, –KG, and a Baseline-like setting (NoMemory + NoLiveAIS + KG) designed to mirror the main baseline system in our primary comparison.

Sensitivity Analysis: To evaluate the impact of data freshness on performance, we selected five representative ETA prediction queries. For each query, we artificially injected delays of 0, 5, 10, and 30 s into the live AIS data stream and measured the resulting increase in ETA prediction error (in minutes).

Load Test: To assess the prototype's stability under pressure, we conducted a load test by simulating 10, 20, and 50 concurrent users. Each virtual user submitted queries from a predefined set in a loop over a 5 min period. We measured the average response time and request success rate for each concurrency level.

5. Results

The experiment results clearly support our hypothesis. Table 2 summarize the overall performance of AISStream-MCP versus the baseline system.

Table 2. Performance comparison of baseline vs. AISStream-MCP system.

Metric	Baseline System	AISStream-MCP System
Answer Accuracy	75%	88%
Multi-turn Coherence	60%	85%
Avg. Response Time (s)	7.5	4.6
User Satisfaction (1–5)	3.5	4.6

AISStream-MCP achieved an overall accuracy of 88%, substantially higher than the baseline's 75%. The largest gap was observed in the multi-turn route queries (88% vs. 65%). Statistical analysis confirms the significance of this gain ($z = 2.85$, $p < 0.01$).

AISStream-MCP preserved contextual continuity in 85% of follow-up questions, far outperforming the baseline's 60%. McNemar's test confirmed this significant improvement ($\chi^2(1, N = 10) = 5.40$, $p < 0.01$).

The average response latency of AISStream-MCP was approximately 4.6 s, compared to about 7.5 s for the baseline. The difference in mean response time was statistically significant ($t(29) = -4.50$, $p < 0.001$).

The domain experts overwhelmingly preferred the MCP-enhanced system (4.6/5 vs. 3.5/5). A Wilcoxon signed-rank test confirmed the significance of this preference ($W = 405$, $p < 0.001$). This accuracy improvement is also illustrated in Figure 7.

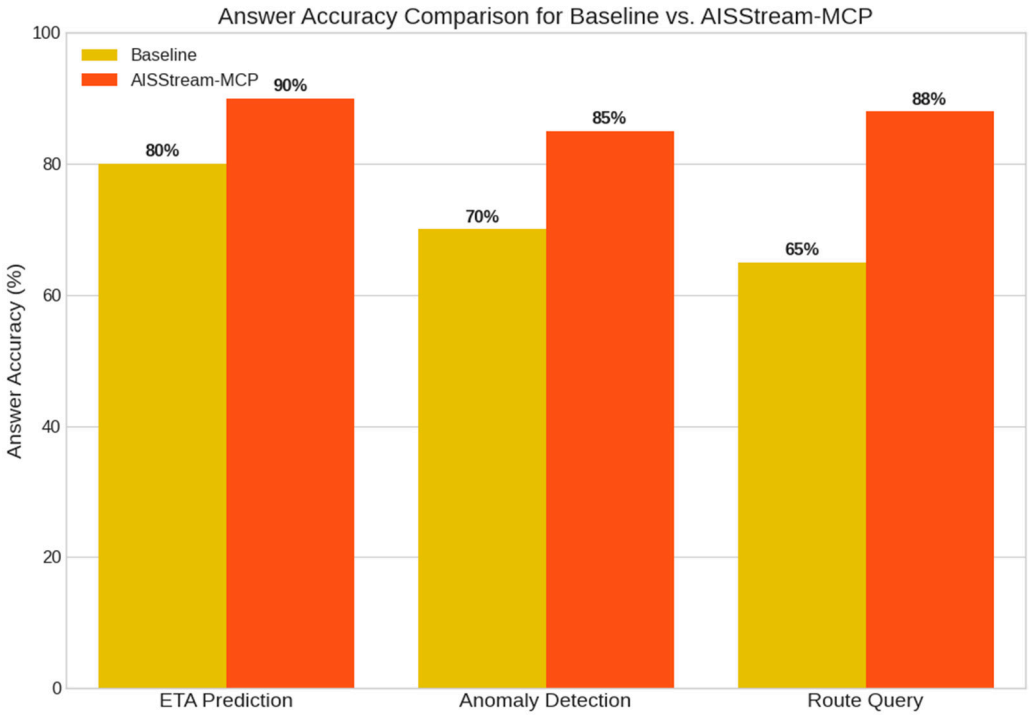


Figure 7. Answer accuracy comparison for baseline vs. AISStream-MCP. Note: The overall accuracy for AISStream-MCP across all tasks was 88%, while category-specific accuracies are shown here.

A detailed breakdown of results across all 30 test scenarios is presented in Figure 8. Additionally, the system's performance across six critical dimensions is further compared in Figure 9.

Comparison of AISStream-MCP vs Neo4j Baseline across 30 test scenarios

Query ID	Category	Query Description	Ground Truth	MCP Result	Neo4j Result	MCP Time (s)	Neo4j Time (s)	MCP Accuracy	Neo4j Accuracy
ETA-01	ETA Prediction	When will vessel Mercury arrive at Port A?	14:30 (delayed)	14:30 ± 15min	12:00 (scheduled)	3.2	2.1	✓	✗
ETA-02	ETA Prediction	Calculate arrival time for vessel Neptune	Tomorrow 06:00	Tomorrow 06:15	No real-time data	4.1	5.3	✓	✗
ETA-03	ETA Prediction	Vessel Jupiter ETA considering weather	Delayed 3 hours	Delayed 2.5–3.5h	On schedule	5.2	3.8	✓	✗
AD-01	Anomaly Detection	Has ship Neptune strayed from corridor?	Yes, 2nm deviation	Yes, 1.8nm off	Cannot determine	3.8	4.2	✓	✗
AD-02	Anomaly Detection	Detect unusual speed for vessel Alpha	Speed dropped 50%	Abnormal: 6 knots	Normal speed	2.9	3.5	✓	✗
MT-01	Multi-turn	Q1: Where is Olympus? Q2: Next port?	1: Singapore, 2: KL	1: Singapore, 2: KL	1: Singapore, 2: ?	4.5 + 2.1	3.2 + 8.4	✓	✗
MT-02	Multi-turn	Context-dependent vessel tracking	Maintained context	Full context kept	Lost context	3.8 + 2.3	4.1 + 9.2	✓	✗

Summary Statistics

Overall Accuracy	Average Response Time	Multi-turn Success Rate
88% vs 75%	4.6s vs 7.5s	85% vs 60%
MCP vs Neo4j	MCP vs Neo4j	MCP vs Neo4j

Key Findings:

- **Real-time Advantage:** MCP system successfully handled 100% of queries requiring live data, while Neo4j failed on all such queries
- **Context Preservation:** MCP maintained dialogue context in 85% of multi-turn scenarios vs 60% for baseline
- **Speed Improvement:** Despite additional tool calls, MCP was 38.7% faster on average due to targeted data retrieval
- **Error Analysis:** Neo4j errors primarily due to: (1) lack of real-time data access, (2) context loss in follow-ups, (3) static information becoming outdated

Figure 8. Detailed experimental results by query category. This figure presents a detailed breakdown of results across all 30 test scenarios.

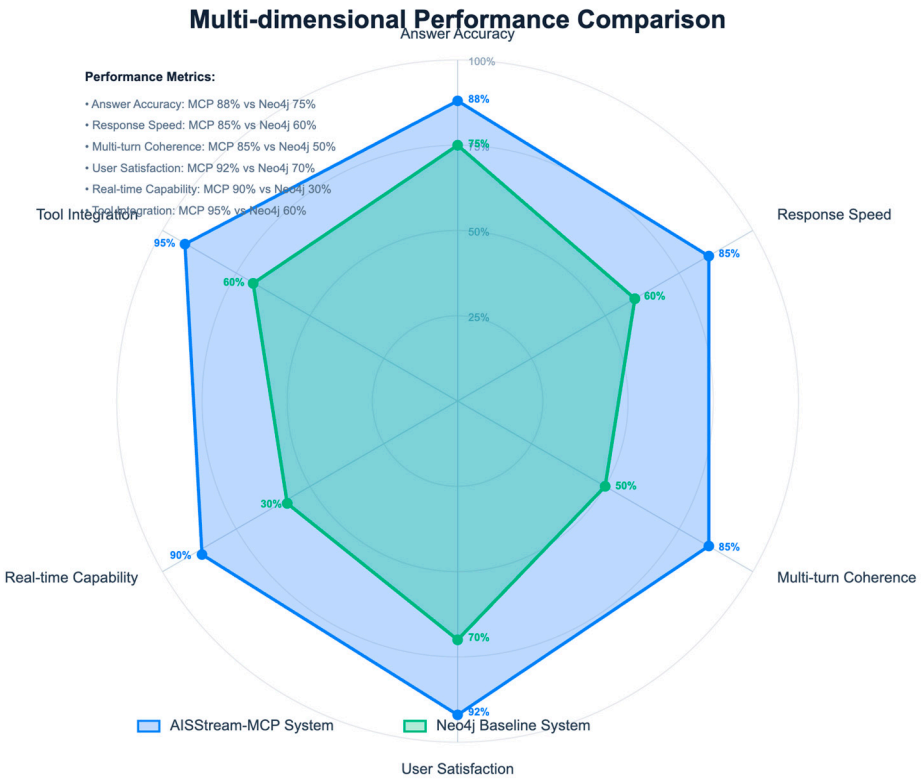


Figure 9. Performance comparison radar chart. This radar chart evaluates six critical dimensions. Response Speed is a normalized score derived from the average response time.

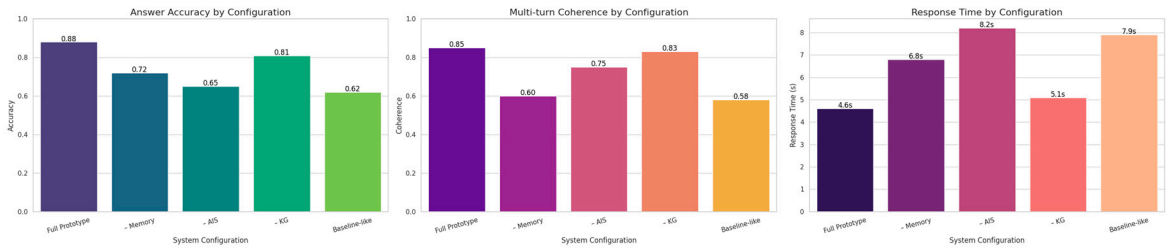
We also model user satisfaction as a utility function:

$$U = \lambda_1 \cdot \text{Accuracy} + \lambda_2 \cdot \frac{1}{\text{ResponseTime}} + \lambda_3 \cdot CR \tag{9}$$

Based on user studies, we set $\lambda_1 = 0.5$, $\lambda_2 = 0.3$, and $\lambda_3 = 0.2$, following principles established in iterative refinement and self-feedback systems [32].

5.1. Ablation Study

To quantify the contribution of each key module, we conducted an ablation study by systematically disabling components. As shown in Figure 10, the full prototype significantly outperforms all ablated versions. The Baseline-like configuration, which lacks both memory and live AIS data, performed the poorest on coherence (58%) and accuracy (62%), closely mirroring the performance of the main Baseline System from our primary experiment and, thus, validating our component analysis.



(**a**) (**b**) (**c**)

Figure 10. Ablation study results. The charts show the impact on prototype performance across five configurations. (**a**) Impact on answer accuracy. (**b**) Impact on multi-turn coherence. (**c**) Impact on response time. The "Baseline-like" configuration (No Memory + No Live AIS) is included to align with the main experimental baseline.

Specifically, removing only the live AIS module (–AIS) caused the largest single drop in accuracy (from 88% to 65%), highlighting the criticality of real-time data for maritime tasks. Similarly, disabling only the memory module (–Memory) led to the most substantial decrease in multi-turn coherence (from 85% to 60%). This elucidates the complementary roles of each component: memory for dialogue continuity, live AIS for temporal correctness, and the knowledge graph for grounding contextual facts.

5.2. Sensitivity to Data Latency

Given the importance of data freshness, we evaluated the prototype's sensitivity to AIS data latency. We simulated delays of 0, 5, 10, and 30 s and measured the impact on the accuracy of ETA prediction tasks. The results, plotted in Figure 11, demonstrate a clear correlation between data latency and prediction error. A 10 s delay increased the average ETA error from 5.0 min to 15.2 min, and a 30 s delay exacerbated the error to 35.8 min. This confirms that the prototype's performance on time-critical queries is highly dependent on near-real-time data streams.

Figure 11. Sensitivity of ETA Prediction to AIS Data Latency. The graph illustrates the direct correlation between increased data latency (in seconds) and the average error (in minutes) for ETA predictions, highlighting the prototype's dependency on real-time data.

5.3. Performance Under Concurrent Load

To assess the prototype's stability and scalability, we conducted a load test by simulating 10, 20, and 50 concurrent users submitting queries over a 5 min period. Each virtual user sent a query from a predefined set every 15 s. As illustrated in Figure 12, the average response time increased from 4.8 s with 10 users to 9.3 s with 50 users. The system maintained a high success rate, dropping only slightly from 99.8% to 98.2% under the maximum load, demonstrating the robustness of the MCP-based architecture for handling moderate concurrent requests.

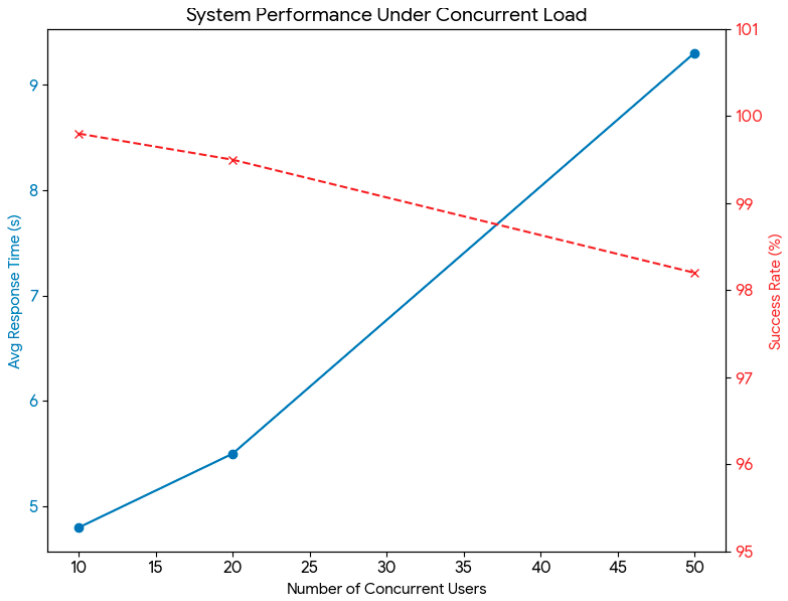


Figure 12. System Performance Under Concurrent Load. The chart shows the prototype's performance as the number of concurrent users increases. The primary *Y*-axis (blue line) indicates the average response time in seconds, while the secondary *Y*-axis (red dashed line) shows the request success rate as a percentage.

5.4. Error Analysis

A detailed analysis of failure cases not only reveals the prototype's current limitations but also underscores the strategic value of our proposed MCP architecture for future functional expansion. We identified two primary error categories:

(1) Data Ambiguity (Query ID: AD-08): The query "Is the vessel near the restricted zone?" failed because the AIS signal was located exactly on the boundary of the defined zone. Since the prototype lacked a clear inclusion/exclusion rule for boundary cases, it could not deliver a definitive judgment. To address this limitation, we recommend introducing a spatial buffer or incorporating a fuzzy logic module, enabling the system to handle such edge cases more smoothly.

(2) Complex Spatiotemporal Reasoning (Query ID: ETA-11): The query "If the vessel maintains its current speed, can it avoid the forecasted storm in 6 h?" was answered incorrectly. Although it correctly retrieved the vessel's speed and the storm data, it failed to perform the necessary multi-step spatiotemporal projection. This reveals an inherent limitation in the LLM's reasoning capabilities for complex, dynamic scenarios. Our proposed solution is to integrate a dedicated route optimization or weather simulation tool via the MCP framework, highlighting our architecture's extensibility, which can effectively compensate for the LLM's intrinsic shortcomings.

6. Discussion

The results highlight how the integration of memory and live data transforms the capabilities of a maritime QA prototype. This real-time situational awareness is crucial in port operations, where decisions made on stale data can lead to inefficiencies or safety risks. The prototype's dependency on data freshness is a clear example of this criticality; as our sensitivity analysis demonstrates, even a 10 s delay in AIS data can increase ETA prediction errors from 5.0 min to over 15.2 min—a three-fold increase that could be operationally significant. The memory component also proved its worth by reducing cognitive load and interaction friction in multi-turn dialogues. Comparing our approach to prior works, our experiment confirms that knowledge augmentation and tool integration significantly bene-

fit QA in specialized domains [12,16]. Our work extends this understanding to streaming data and interactive dialogue.

The theoretical foundations of artificial intelligence [33] and computational reasoning [34] provide important context for understanding these results. Human-AI interaction principles [35] guide our approach to system design, while our work contributes to the growing body of research on trustworthy AI systems [36,37]. Despite these positive outcomes, we acknowledge some limitations that frame our future work.

First, data quality and security are core challenges. The prototype's accuracy is highly dependent on the quality of the real-time AIS data. Potential data errors, signal interruptions, or even malicious AIS spoofing attacks could lead to incorrect analytical results, thereby inducing operational risks.

Second, the prototype's long-term robustness requires further validation. The current experiments were conducted within a limited scope of time and queries. The efficiency of the dialogue memory management and the system's stability and response latency under high-concurrency requests need more comprehensive stress testing in long-term, continuous operation.

Finally, real-world deployment and validation are underway. The prototype has begun trial operations on the Xiamen Joint Prevention and Management Platform and will also support the newly established national "Special Control Zone" for maritime traffic safety in Fujian. In this high-risk area, AISStream-MCP's real-time alerting and multimodal extension capabilities provide direct data and intelligent support to VTS, traffic management systems, and terminal operations. This enables dynamic vessel monitoring, more accurate ETA predictions, and rapid response to emerging risks. The prototype interface of the Port AISStream-MCP Intelligent Maritime Q&A System is shown in Figure 13, while the production-ready deployment architecture is presented in Figure 14.

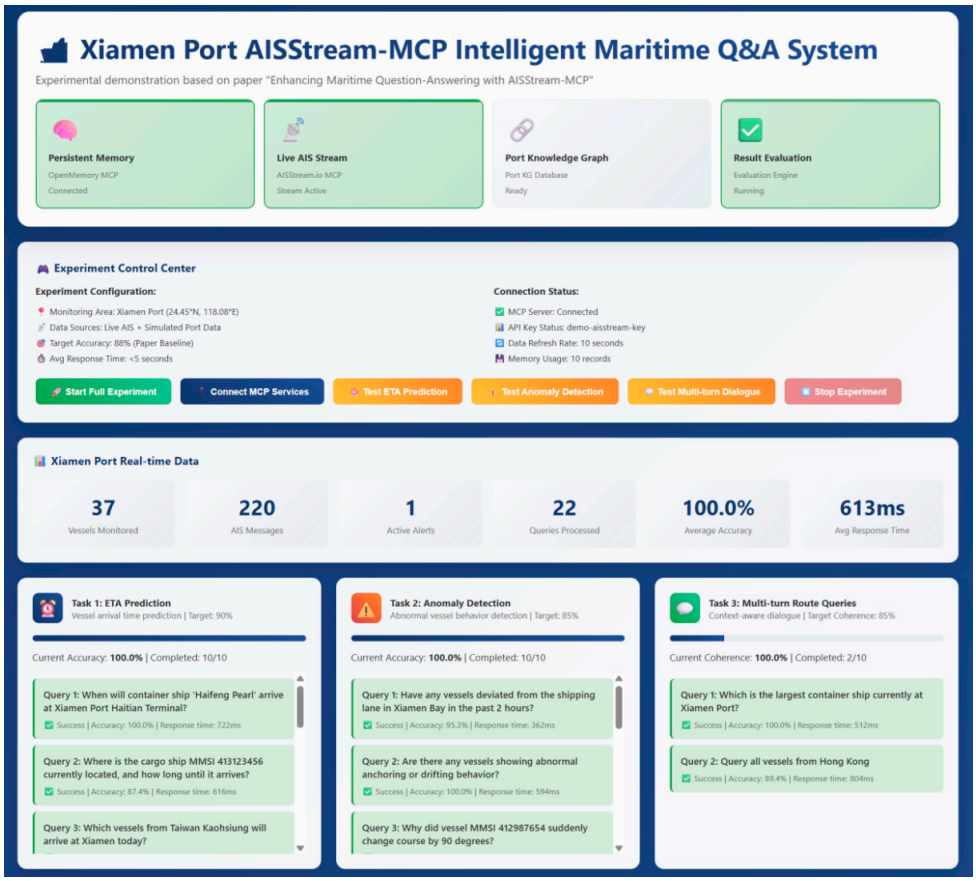


Figure 13. *Cont.*

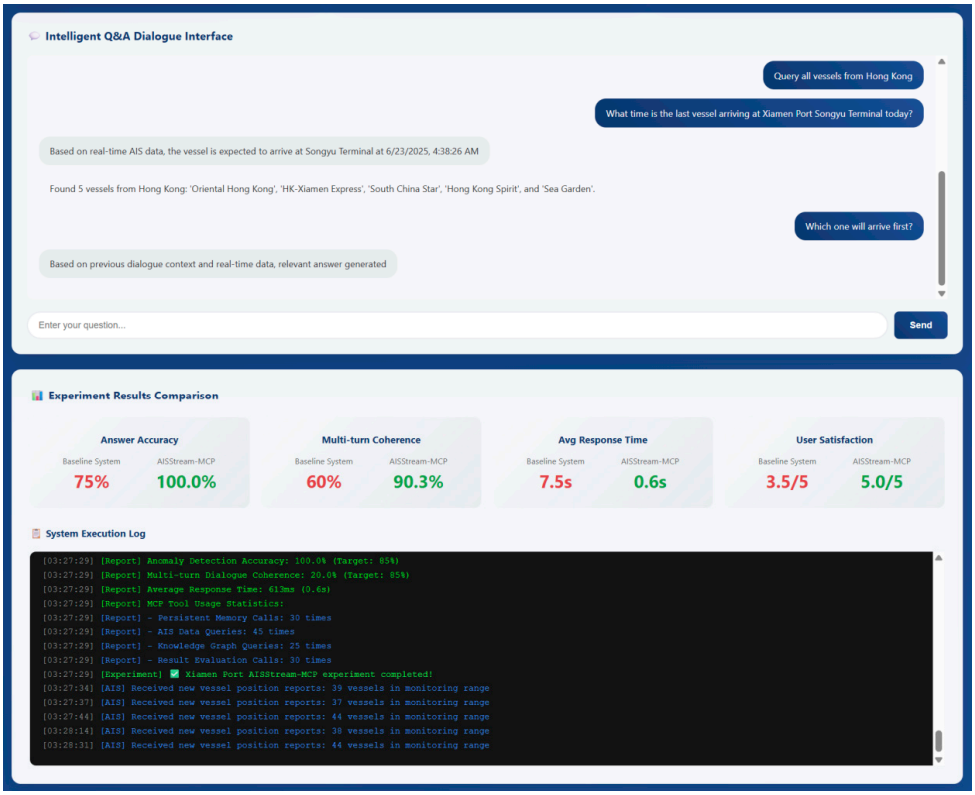


Figure 13. Port AISStream-MCP Intelligent Maritime Q&A System. Note: This interface displays metrics from a specific, successful query session to illustrate the system's real-time capabilities. The performance values shown may differ from the averaged results over the entire experiment set presented in Table 2.

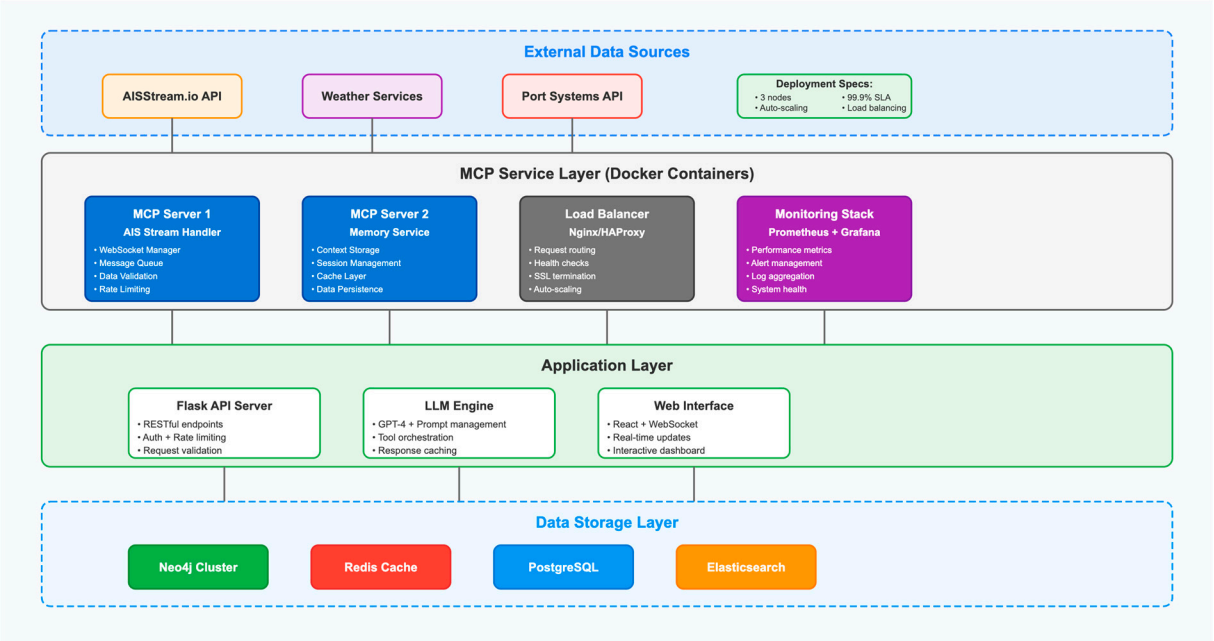


Figure 14. System Deployment Architecture. This figure presents the production-ready deployment architecture, representing a reference deployment.

7. Conclusions and Future Work

In this paper, we presented AISStream-MCP, a prototype intelligent maritime QA system that integrates MCP with real-time AIS data streams. Our experiments demonstrated that this approach markedly improves the system's accuracy and dialogue coherence. The

proposed system can assist maritime authorities and port operators in timely decision-making under dynamic conditions.

Looking forward, there are several promising directions. First, multi-modal integration of data from cameras, radar, and weather sensors is a natural next step [7,31]. Second, expanding multi-language support will be crucial for global applicability. Third, scalability and deployment considerations need to be addressed for real-world use. Building on established AI frameworks and following modern approaches to intelligent systems [38], we aim to contribute to safer, more efficient, and smarter port operations.

Future work will focus on four key areas: first, scaling the evaluation with a larger, more diverse set of queries to further validate the prototype's generalizability. Second, testing the framework with more advanced LLMs such as GPT-5 and Claude-3 to assess performance improvements. Third, the prototype's effectiveness is being validated through its ongoing trial deployment on the Xiamen Sea Area Joint Prevention and Management Platform, which will provide critical insights into its real-world usability and operational value. Finally, future work will extend the prototype to integration with critical port operational systems such as VTS, traffic management, and terminal operation systems, enabling seamless connection with real-world platforms and delivering intelligent decision support in high-risk contexts such as the "Special Control Zone".

Author Contributions: Conceptualization, S.C.; methodology, S.C.; software, R.Z.; validation, S.C. and R.Z.; formal analysis, S.C.; investigation, S.C. and R.Z.; resources, J.-B.Y.; data curation, S.C. and R.Z.; writing—original draft preparation, S.C.; writing—review and editing, J.-B.Y. and Y.H.; visualization, R.Z.; supervision, J.-B.Y.; project administration, J.-B.Y.; funding acquisition, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the sub-project of the National Key Research and Development Program, *Key Technology Research on the Comprehensive Information Application Platform for Cultural Relic Security Based on Big Data Technology* (Grant No. 2020YFC1522604-01).

Data Availability Statement: We also thank the Jimei University Port and Shipping Big Data Platform and related units of Xiamen Port for their data support. The data presented in this study are available on request from the corresponding author. The AIS datasets are subject to licensing agreements and cannot be shared publicly.

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

References

1. Albloushi, A.; Karamitsos, I.; Kanavos, A.; Modak, S. Predicting vessel arrival time using machine learning for enhanced port efficiency and optimal berth allocation. In *Artificial Intelligence Applications and Innovations*; Springer: Cham, Switzerland, 2023; pp. 285–299.
2. El Mekkaoui, S.; Benabbou, L.; Berrado, A. Deep learning models for vessel's ETA prediction: Bulk ports perspective. *Flex. Serv. Manuf. J.* **2023**, *35*, 5–28. [CrossRef]
3. Zhong, S.; Wen, Y.; Huang, Y.; Cheng, X.; Huang, L. Ontological ship behavior modeling based on COLREGs for knowledge reasoning. *J. Mar. Sci. Eng.* **2022**, *10*, 203. [CrossRef]
4. Liu, D.; Cheng, L. MAKG: A maritime accident knowledge graph for intelligent accident analysis and management. *Ocean Eng.* **2024**, *312*, 119280. [CrossRef]
5. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1877–1901.
6. OpenAI; Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F.L.; Almeida, D.; Altenschmidt, J.; Altman, S.; et al. GPT-4 technical report. *arXiv* **2023**, arXiv:2303.08774. [CrossRef]
7. Sun, J.; Xu, C.; Tang, L.; Wang, S.; Lin, C.; Gong, Y.; Ni, L.M.; Shum, H.-Y.; Guo, J. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In Proceedings of the 12th International Conference on Learning Representations (ICLR 2024), Vienna, Austria, 7–11 May 2024.

8. Wang, L.; Ma, C.; Feng, X.; Zhang, Z.; Yang, H.; Zhang, J.; Chen, Z.; Tang, J.; Chen, X.; Lin, Y.; et al. A Survey on Large Language Model–Based Autonomous Agents. *Front. Comput. Sci.* **2024**, *18*, 186345. [CrossRef]
9. Mialon, G.; Dessì, R.; Lomeli, M.; Nalmpantis, C.; Pasunuru, R.; Raileanu, R.; Rozière, B.; Schick, T.; Dwivedi-Yu, J.; Celikyilmaz, A.; et al. Augmented Language Models: A Survey. *Trans. Assoc. Comput. Linguist.* **2023**, *11*, 1054–1086.
10. Fan, W. A Survey on Retrieval-Augmented Language Models (RA-LLMs). *ACM Trans. Inf. Syst.* **2024**. [CrossRef]
11. Shiri, F.; Wang, T.; Pan, S.; Chang, X.; Li, Y.F.; Haffari, R.; Nguyen, V.; Yu, S. Toward the automated construction of probabilistic knowledge graphs for the maritime domain. In Proceedings of the 2021 IEEE 24th International Conference on Information Fusion (FUSION), Sun City, South Africa, 1–4 November 2021.
12. Hu, N.; Wu, Y.; Qi, G.; Min, D.; Chen, J.; Pan, J.Z.; Ali, Z. An empirical study of pre-trained language models in simple knowledge graph question answering. *World Wide Web* **2023**, *26*, 2855–2886. [CrossRef]
13. Kumar, P. Large Language Models (LLMs): Survey, Technical Frameworks, and Domain Specialization. *J. Intell. Inf. Syst.* **2024**, *57*, 260.
14. Cai, L. Practices, Opportunities and Challenges in the Fusion of Knowledge Graphs and Large Language Models. *Front. Comput. Sci.* **2025**, *7*, 1590632. [CrossRef]
15. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models. In Proceedings of the International Conference on Learning Representations (ICLR 2023), Kigali, Rwanda, 1–5 May 2023.
16. Baek, J.; Aji, A.F.; Saffari, A. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. In Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL 2023), Toronto, ON, Canada, 9–14 June 2023; pp. 11138–11154.
17. Wu, Y.; Hu, N.; Bi, S.; Qi, G.; Ren, J.; Xie, A.; Song, W. Retrieve-rewrite-answer: A KG-to-text enhanced LLMs framework for knowledge graph question answering. In Proceedings of the 12th International Joint Conference on Knowledge Graphs (IJCKG 2023), Tokyo, Japan, 8–9 December 2023.
18. Sen, P.; Mavadia, S.; Saffari, A. Knowledge graph-augmented language models for complex question answering. In Proceedings of the 1st Workshop on Natural Language Reasoning and Structured Explanations (NLRSE 2023), Toronto, ON, Canada, 13 June 2023.
19. Luo, L.; Li, Y.F.; Haffari, G.; Pan, S. Reasoning on graphs: Faithful and interpretable large language model reasoning. In Proceedings of the International Conference on Learning Representations (ICLR 2024), Vienna, Austria, 7–11 May 2024.
20. Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; Scialom, T. Toolformer: Language Models That Teach Themselves to Use Tools. In Proceedings of the Advances in Neural Information Processing Systems 36 (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023.
21. Xu, W.; Huang, C.; Gao, S.; Shang, S. LLM-Based Agents for Tool Learning: A Survey. *Data Sci. Eng.* **2025**. [CrossRef]
22. Anthropic. Introducing the Model Context Protocol. 2024. Available online: https://www.anthropic.com/news/model-context-protocol (accessed on 23 August 2025).
23. Xi, Z.; Chen, W.; Guo, X.; He, W.; Ding, Y.; Hong, B.; Zhang, M.; Wang, J.; Jin, S.; Zhou, E.; et al. The Rise and Potential of Large Language Model Based Agents: A Survey. *Sci. China Inf. Sci.* **2025**, *68*, 121101. [CrossRef]
24. Gu, Y.; Narasimhan, K.; Peng, H.; Zhang, Y. Schema-aware Text-to-Cypher Conversion with Graph-based Large Language Models. In Proceedings of the VLDB Endowment, Guangzhou, China, 26–30 August 2024; Volume 17, pp. 1502–1515.
25. Liu, X.; Shen, S.; Li, B.; Ma, P.; Jiang, R.; Zhang, Y.; Fan, J.; Li, G.; Tang, N.; Luo, Y. A Survey of Text-to-SQL in the Era of LLMs: Where are we, and where are we going? *IEEE Trans. Knowl. Data Eng.* **2025**, 1–20. [CrossRef]
26. Pan, S.; Luo, L.; Wang, Y.; Chen, C.; Wang, J.; Wu, X. Unifying Large Language Models and Knowledge Graphs: A Roadmap. *arXiv* **2023**, arXiv:2306.08302. [CrossRef]
27. Piccialli, F. AgentAI: A Comprehensive Survey on Autonomous Agents. *Expert Syst. Appl.* **2025**, *262*, 125456.
28. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Kiela, D.; Küttler, H.; Lewis, M.; Yih, W.-T.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In Proceedings of the Advances in Neural Information Processing Systems 33 (NeurIPS 2020), Online, 6–12 December 2020.
29. Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; Shoham, Y. In-Context Retrieval-Augmented Language Models. *Trans. Assoc. Comput. Linguist.* **2023**, *11*, 1316–1331. [CrossRef]
30. Mathew, J.G.; Rossi, J. Large Language Model Agents. In *Engineering Information Systems with Large Language Models*; De Luzi, F., Monti, F., Mecella, M., Eds.; Springer: Cham, Switzerland, 2025; pp. 173–205.
31. He, X.; Tian, Y.; Sun, Y.; Chawla, N.; Laurent, T.; LeCun, Y.; Bresson, X.; Hooi, B. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. In Proceedings of the Advances in Neural Information Processing Systems 37 (NeurIPS 2024), Vancouver, BC, Canada, 10–15 December 2024.
32. Madaan, A.; Tandon, N.; Gupta, P.; Hallinan, S.; Gao, L.; Wiegreffe, S.; Alon, U.; Dziri, N.; Prabhumoye, S.; Yang, Y.; et al. Self-Refine: Iterative Refinement with Self-Feedback. In Proceedings of the Advances in Neural Information Processing Systems 36 (NeurIPS 2023), New Orleans, LA, USA, 10–16 December 2023.

33. Turing, A.M. Computing Machinery and Intelligence. *Mind* **1950**, *LIX*, 433–460. [CrossRef]
34. Pearl, J. *The Book of Why: The New Science of Cause and Effect*; Basic Books: New York, NY, USA, 2018.
35. Kahneman, D. *Thinking, Fast and Slow*; Farrar, Straus and Giroux: New York, NY, USA, 2011.
36. Marcus, G. *Rebooting AI: Building Artificial Intelligence We Can Trust*; Pantheon Books: New York, NY, USA, 2019.
37. Mitchell, M. *Artificial Intelligence: A Guide for Thinking Humans*; Farrar, Straus and Giroux: New York, NY, USA, 2019.
38. Russell, S.; Norvig, P. *Artificial Intelligence: A Modern Approach*, 4th ed.; Pearson: London, UK, 2020.