Designing the AI Web: Architectural Principles for Interoperable Artificial Intelligence Systems.

Abstract:

With the field of artificial intelligence undergoing constant evolution, AI systems are rapidly shifting from single models to complex networks that involve countless specialised agents. This transition is supported by several studies that have established multi-agent systems to offer advanced capabilities when compared to single LLM-powered agents. While it is aimed for these to work together by interacting with one another, the absence of a standard structure and mode of communication proves to be a challenge.

This paper introduces and elaborates on the idea of an "AI Web," a unified network of interoperable artificial intelligence agents capable of collaborating seamlessly across numerous diverse frameworks and applications. We draw inspiration from the internet, where machines use certain protocols for communication. Building upon previous studies, we propose a nine-layer structure embedded with Model Context Protocol (MCP) and Agent-to-Agent (A2A) protocols to allow effective communication between agents.

The model was conceptually evaluated through application scenarios in enterprise and business domains. The analysis suggested that standardised communication improves coordination and collaboration between AI agents, reducing integration problems and thereby resulting in overall system efficiency. The study argues that the next generation of artificial intelligence will depend not only on developing increasingly capable individual agents but also on establishing universal architectural principles and communication standards that allow them to function as part of a connected ecosystem.

Keywords: Artificial Intelligence Web, Systems Interoperability, Model Context Protocol (MCP), Multi-Agent Architecture, Semantic Systems, Distributed Intelligence.

Chapter 1: Introduction

1.1 Evolution of Artificial Intelligence

Artificial Intelligence has seen numerous phases since its birth. Systems were first constructed utilising symbolic reasoning and rule bases hand-coded as so-called expert systems. These systems worked well in structured environments, but it was difficult to adapt them to unexpected scenarios (Russell & Norvig, 2020). The area then progressed to statistical and machine-learning methods where judgements are inferred from data rather than written out in advance, allowing systems to grow more flexible with exposure to more cases. Modern systems carry on that data-driven trajectory but at a scale that earlier researchers could not have predicted, learning from images, sounds and text to detect complex links and make judgements in dynamic situations. This trend can be described as a journey from rule-based expert systems, via statistical machine learning and deep learning, to transformer-based large language models, and now autonomous multi-agent systems.

1.2 Rise of Large Language Models

The introduction of the Transformer architecture, built around a self-attention mechanism that lets a model weigh many parts of its input simultaneously, was a turning point for the field (Vaswani et al., 2017). It made it practical to train language models on web-scale text, giving rise to systems capable of broad language understanding and in-context, few-shot learning across tasks such as summarisation, translation, question answering, code generation, and logical reasoning, without task-specific retraining. Foundation-model research since then has repeatedly shown that a single large, pre-trained model can be adapted to a wide range of downstream tasks rather than being built from scratch for each one (Bommasani et al., 2021; Schneider et al., 2024).

1.3 Transition from Standalone AI to AI Ecosystems

Artificial intelligence is now entering a phase in which systems are expected to work together rather than in isolation. The traditional pattern of a single input producing a single output is giving way to architectures in which multiple AI agents cooperate, delegate sub-tasks, and call external tools to achieve a shared goal (Wooldridge, 2020). Recent information-systems research frames this shift as “multi-agent AI” (MAAI): ensembles of AI-based agents whose interactions are not fixed in advance but emerge dynamically in response to context, extending decades of multi-agent systems research into contemporary, foundation-model-based settings (Allmendinger et al., 2026).

1.4 Need for Interconnected AI

Many real-world problems remain too complex to be solved efficiently by a single model, since they require multiple forms of expertise, continuous planning, external tool usage, and long-term memory, particularly in scientific research, software engineering, healthcare, decision-making, and enterprise automation. Expecting one model to perform all of these functions introduces practical limits on scalability, reliability, and maintainability. No single model can internalise the

whole of human knowledge or every specialised operational task; getting the most out of machine intelligence therefore means moving away from isolated instances and toward integration, so that a legal-analysis model, a real-time sensor-processing pipeline, and a customer-facing assistant can share context and results (Horvitz, 2022). In this way, the intelligence of one system becomes usable by others, and such systems build on one another's capabilities rather than duplicating them.

1.5 Problem Statement

1.5.1 AI Systems Operate in Silos

AI capability keeps improving, but the ability of different systems to work together has not kept pace. Large technology providers tend to keep their AI stacks closed, which restricts how far outside developers can extend or recombine them. The systems themselves are often highly capable, but they are designed to work within one company's ecosystem, which limits how flexibly users can apply AI across different contexts.

1.5.2 Limited Collaboration Between AI Platforms

Because these systems are not structurally compatible, real-time collaboration between models on different platforms is inefficient. A model on one platform cannot hand off a sub-task to a model on a competing platform without a bespoke, brittle integration being built first. This limits the exchange of context and state between systems and makes it hard to form ad hoc computational networks on demand.

1.5.3 Lack of Universal Communication Protocols

The classic internet abstracted away hardware differences through universal standards such as TCP/IP and HTTP. The AI world does not yet have an equivalent. Communication today is mostly performed via proprietary REST endpoints and inconsistent JSON payloads, with no standard semantic schemas, machine-readable capability descriptions or common session-management rules, and it is exactly this gap that emerging protocols such as MCP and A2A seek to close (Hughes et al., 2025).

1.5.4 Fragmented AI Ecosystems

This fragmentation raises operational friction and creates vendor lock-in, since enterprises are often pushed toward a single vendor's stack, reducing their flexibility. Beyond commercial settings, the absence of interoperability is also a barrier wherever secure, cross-platform data exchange and coordination are required, as in healthcare, scientific research, and public infrastructure, where interoperability is an architectural requirement rather than a convenience.

1.6 Research Objectives

- Examine the concept of the AI Web: define its core structural paradigms, requirements, and boundaries as an open, globally distributed network of machine intelligence.

- Identify the architectural principles – modularity, loose coupling, and separation of state – that make AI interoperability possible.
- Specify the layers needed for global AI interoperability, from physical hardware up through coordination and governance.
- Study the communication protocols enabling AI-to-AI collaboration, including MCP, gRPC, A2A, and semantic routing.
- Set out future research directions: standardisation gaps, safety research, and the technical obstacles to a sustainable, open agent network.

1.7 Research Questions

- What is the AI Web, and how does it differ from today's web infrastructure and from centralised AI systems?
- What architectural principles must software engineers enforce so that diverse AI systems achieve interoperability without losing contextual accuracy?
- What architectural layers are needed for a robust, worldwide network of autonomous AI agents?
- Which communication protocols and semantic structures are most effective for machine-to-machine collaboration?
- What technical, structural, and regulatory challenges act as barriers to global integration of AI systems?

1.8 Scope

This paper covers the software architectures, data structures, semantic technologies, and communication protocols that make interoperability possible between AI systems on different platforms, including how context, goals, state, and payloads move across organisational boundaries. Underlying computational hardware (GPU or TPU clusters) is mentioned only where it bears directly on the architecture; hardware microarchitecture and fabrication are out of scope.

1.9 Limitations

Because engineering standards for the AI Web are still evolving quickly, long-run empirical performance data for protocols such as MCP remains limited compared with mature web protocols. This study also relies on secondary academic and industry documentation available up to 2026. Since many frontier AI systems remain closed-source, some structural claims rest on architectural whitepapers and open equivalents rather than direct inspection of proprietary systems.

Chapter 2: Theoretical and Conceptual Framework

2.1 Artificial Intelligence Ecosystems

An AI ecosystem is a heterogeneous environment of models, data sources, computing infrastructure, and other entities that interact dynamically within a shared virtual space (Jennings, 2001). Viewed this way, an AI model is not a static transformer but a node in a network that exchanges context, adapts its behaviour to its environment, and shares computational load with its neighbours.

2.2 AI Interoperability

AI interoperability is the ability of two or more independently built AI systems potentially using different base architectures, such as transformers versus state-space models, and different data schemas to exchange data and capabilities securely, without a human in the loop. Following standard systems-engineering practice, this can be broken into three dimensions, summarised in Table 1.

Interoperability Dimension	Technical Requirement within AI Ecosystems
Syntactic	Common serialisation formats (JSON, Protocol Buffers) and shared transmission media.
Semantic	A shared understanding of data, context, intent, and agent capability, grounded in explicit ontologies.
Pragmatic	Shared alignment on execution goals, task boundaries, constraints, and policy enforcement.

Table 1. Dimensions of AI interoperability.

2.3 Distributed AI Systems

Distributed Artificial Intelligence (DAI) spreads computational and cognitive work across many independent nodes rather than sending every request to one large data centre (Bond & Gasser, 1988). Breaking a problem into smaller sub-problems that different nodes solve in parallel improves resource use and reduces the risk of a single point of failure.

2.4 Multi-Agent Systems

Multi-Agent Systems (MAS) are the practical, real-world form that DAI takes: a network of agents that each perceive their environment through sensors or APIs, reason about it using mechanisms such as chain-of-thought loops, and act on it (Wooldridge, 2020). Coordinating such a network so that agents do not work against each other remains a central design problem, and recent MAS research increasingly frames this as a design-and-society problem: how to build capable individual agents, and separately, how to organise them into effective collectives.

2.5 Service-Oriented Architecture (SOA)

Service-Oriented Architecture is a foundational pattern for building loosely coupled, discoverable AI components. Under SOA, a component exposes its functionality through a defined interface, so that consumers can use its services without needing to know how it was trained or how its internal parameters work.

2.6 Microservices Architecture

Microservices extend SOA by breaking a large AI workload into smaller, independently deployable parts: one service for vector search, another for context engineering, another for guardrails, and so on, rather than one monolithic service handling all of them together.

API Gateway → Guardrail Microservice → Vector Search Microservice → Inference Microservice → Audit Logging Microservice

This lets individual components be scaled, updated, or replaced without disrupting the wider system (Newman, 2021).

2.7 Knowledge Graphs and Semantic Web

The Semantic Web vision of making machine-readable meaning, not just machine-readable data, travel with information across the web remains the conceptual backbone of AI-to-AI understanding (Berners-Lee et al., 2001). Knowledge graphs operationalise this idea: RDF triples, OWL ontologies, and enterprise knowledge graphs let AI systems move past keyword search toward genuine semantic retrieval, connecting structured records with unstructured content through shared vocabularies. This matters directly for AI systems: knowledge graphs are increasingly treated as a foundation for explainable AI and data interoperability, since combining symbolic structure with data-driven models improves a system's transparency and robustness.

2.8 Communication Standards

2.8.1 APIs (Application Programming Interfaces)

APIs remain the core interface through which software platforms talk to each other, but conventional RESTful APIs tend to use rigid, pre-defined schemas that do not adapt well to dynamic agent negotiation.

2.8.2 Model Context Protocol (MCP)

MCP is an open standard that provides a secure, bi-directional interface between AI models and both data sources and developer tools. It defines standardised JSON-RPC interfaces that let a model discover available tools, inspect data schemas, and retrieve contextual attachments on demand.

2.8.3 Agent-to-Agent (A2A) Communication

The A2A protocol governs how autonomous agents talk directly to one another, defining message formats for abstract goals, task assignment, errors, and state context across long, multi-turn exchanges.

2.8.4 Data Exchange Formats

Low-latency serialisation matters a great deal for AI-to-AI communication. JSON remains dominant at human-facing edges, but Protocol Buffers and Apache Arrow are widely used inside high-throughput agent infrastructure.

2.9 Architectural Layers of the AI Web

Building a reliable, globally scalable AI Web means separating concerns into distinct architectural layers. The blueprint proposed here comprises nine layers, shown in Table 2.

Layer	Function
1. Physical	Compute nodes and networking hardware.
2. Infrastructure	GPUs, cloud, and edge computing resources.
3. Data	Vector databases and metadata lakes.
4. Knowledge	Ontologies and knowledge graphs.
5. Model	LLMs, vision models, and specialised models.
6. Agent	Memory, planning, and reasoning.
7. Communication	Protocols and APIs.
8. Coordination	Orchestration across agents.
9. Governance	Compliance and trust.

Table 2. Nine-layer architecture of the proposed AI Web.

2.10 Communication Protocols

Reaching global interoperability depends on choosing the right protocol for a given need. REST APIs are simple and well understood, but rely on synchronous HTTP calls and fixed endpoints, which suit predictable integrations but are poorly suited to open-ended agent negotiation. GraphQL lets a client fetch exactly the data it needs in one request, avoiding REST's over- and under-fetching problems, though at the cost of a more complex setup. gRPC, built on HTTP/2 and Protocol Buffers, offers high-performance remote procedure calls but no semantic layer of its own.

MCP adds that semantic layer for AI specifically, letting a model discover tools and context dynamically over JSON-RPC. A2A complements this by carrying high-level intentions, conversational state, and task delegation across multi-agent workflows. JSON-RPC underlies much of this as a lightweight, stateless messaging substrate, while event-driven architectures and WebSockets support long-lived, full-duplex connections for real-time state updates. A further idea, semantic communication, goes one step beyond all of these by transmitting not just encoded data but an explicit representation of what that data means.

2.11 Architectural Principles

Modularity: Components must be autonomous and loosely connected such that a change in one model or microservice does not ripple outward.
Scalability: The design should be able to handle changing network conditions and scale resources without performance degradation.
Interoperability: Platforms should rely on open and vendor-neutral protocols, not on proprietary protocols.
Reliable: The network should not change state in the presence of connection failures.
Flexibility: Routeing and task assignment should be done considering different agent availability and operating cost.
Explainability: Agents should retain an auditable log of their reasoning, tool use and decisions.
Privacy: Implement robust data-privacy safeguards with zero-trust verification and cryptographic technologies like zero-knowledge proofs.
Security: Full encryption, mutual authentication (mTLS) and sandboxing to keep the network safe from hostile assaults.
Maintainability: Code and integration layers should be versioned so changes do not silently damage dependent services.
Fault tolerance: The system continues to operate if a remote agent crashes while executing a task. The coordination layer detects the failure and rolls back state and migrates the workflow to a different node.

Chapter 3: Literature Review

Earlier work on software integration and multi-agent coordination laid useful groundwork, but the rise of large language models has exposed real gaps. Early distributed-AI research studied narrow multi-agent systems coordinated through deterministic mechanisms such as the Contract Net Protocol (Smith, 1980); these systems could allocate tasks, but had little semantic flexibility, and any change in input format tended to break the integration. Semantic Web technology later offered a more systematic route to machine-understandable information through ontologies and OWL (Berners-Lee et al., 2001), though hand-built ontologies proved difficult to scale. Service-

oriented and microservices architectures then supplied clean, reusable ways to encapsulate software functionality behind APIs (Newman, 2021), but conventional microservices still leave deep-learning components as isolated boxes with no shared way to exchange planning information, structured knowledge, or memory state (Zaharia et al., 2018).

3.1 Interoperability and Enterprise Information Systems

Enterprise-systems research treats interoperability as a design problem to be solved deliberately, not assumed. Ben Hassen et al. (2025), writing in the Business Process Management Journal, address what they call the "three-fit" barrier in enterprise information systems: vertical, horizontal, and transversal fit problems – and propose an ontology-driven design-science approach to formalise business processes so that different systems can interpret them consistently. Their argument that ontological grounding improves integration and adaptability across an enterprise's information systems maps closely onto the knowledge layer of the architecture proposed here. In a related vein, Aktürk's (2021) bibliometric study of AI in enterprise resource planning finds that AI-ERP research has clustered around techniques such as machine learning and fuzzy logic, but that studies explicitly addressing cross-system AI interoperability remain comparatively scarce a gap this paper's nine-layer framework is intended to help close.

3.2 Multi-Agent AI and Organisational Automation

A recent study in Electronic Markets reframes the emergence of multi-agent AI (MAAI) as a shift from static, rule-based automation toward adaptive systems of interacting, foundation-model-based agents (Allmendinger et al., 2026). The authors propose a five-component framework: foundation model, data-centric perception and action, dynamic orchestration, agent-integrated workflow, and interaction interface – that separates the technical, organisational, and human-facing dimensions of multi-agent automation. This decomposition is broadly compatible with the nine-layer AI Web architecture developed here, though it is scoped to organisational knowledge work rather than global network interoperability. Related work traces how AI-enabled information systems progressively team humans with increasingly autonomous agents in networked business settings, underscoring that autonomy is better understood as a spectrum than a binary property (Hofmann et al., 2024).

3.3 AI Governance and Accountability

As agents gain autonomy, governance becomes as important as capability. Camilleri (2024) reviews ethical considerations in AI governance and argues that organisations still lack consistent principles and regulatory guidance for deploying AI and machine-learning systems responsibly a gap that becomes sharper once multiple autonomous agents, rather than a single model, are making decisions in concert. This concern extends into interoperability scholarship more broadly: cross-border and cross-platform data exchange raises distinct governance questions once agents, rather than humans, are the ones initiating the exchange.

3.4 Semantic Web Foundations for Agent Communication

Fareh (2019) examines how incomplete knowledge in Semantic Web ontologies can be modelled using Bayesian networks, addressing the practical problem that domain ontologies are rarely complete or free of ambiguity. This matters directly for AI-to-AI communication: if the ontologies underpinning semantic interoperability are themselves uncertain, any agent reasoning over them inherits that uncertainty. This probabilistic treatment of ontology gaps offers one technical route toward more robust semantic layers within the AI Web architecture, complementing the more deterministic RDF/OWL treatments common in enterprise knowledge-graph work.

3.5 Cross-Platform Agent Protocols

Information-systems scholarship has also begun to examine agentic systems directly. Hughes et al. (2025) provide a multi-expert analysis of AI agents and agentic systems, while a companion study considers how agentic systems are reshaping global IT management practice. Both converge on a similar point: organisations adopting agentic AI face a widening gap between the pace of technical capability and the maturity of the standards, protocols, and governance structures needed to manage it safely across platforms. This is reinforced from the software-engineering side by Liu et al. (2025), who catalogue architectural patterns for foundation-model-based agents and note that design patterns proven within a single agent rarely transfer cleanly to multi-agent, cross-platform settings without an explicit interoperability layer.

3.6 Research Gap

Across this literature, communication protocols, semantic graphs, agent governance, and enterprise integration are generally studied as separate concerns. What is comparatively rare is a single architectural account that connects low-level network protocols to high-level semantic reasoning and safety governance within one coherent stack. This paper's nine-layer framework is an attempt to close that gap by treating these strands as layers of one system rather than as separate literatures.

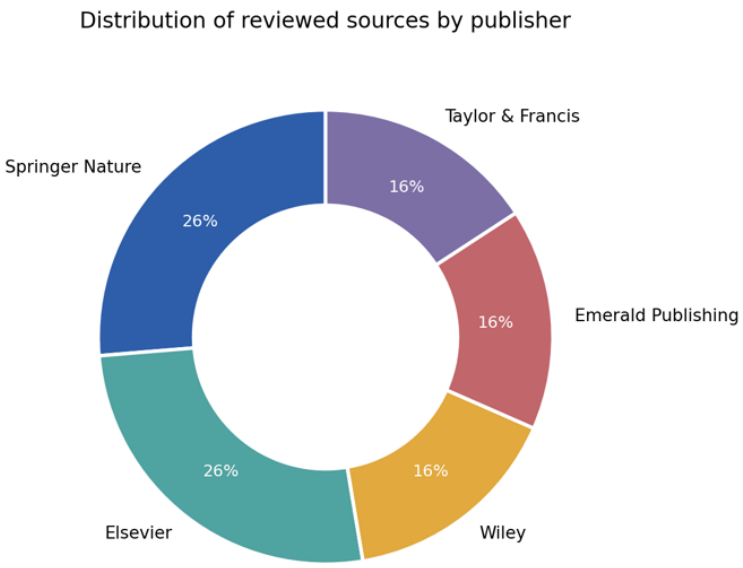

Figure 1. Distribution of the peer-reviewed sources underpinning this literature review, by publisher.

Chapter 4: Case Studies

Case Study 1: AI Assistants Interfacing with Productivity Platforms
Context: Corporate AI assistants are being used more and more by companies to manage calendars, create content and coordinate processes across platforms like Microsoft 365 and Google Workspace.
Architecture: Microservice architecture with agents per workplace that communicate to platforms via webhooks and API gateways.
Interoperability approach: OAuth 2.0 authentication and JSON payload for corporate schema
Benefits: Reduced administrative overhead due to automatic scheduling and cross-platform notifications.
Challenges: Brittle integrations when underlying platform APIs change; loss of conversational context as information traverses platforms.
Lesson: Full RESTful integration is too stiff for dynamic, multi-platform operations. The API layer needs a semantic layer on top for the agents.

Case Study 2: Model Context Protocol (MCP) Enabling Tool Integration
Context. A developer assistant should be capable of inspecting database schemas, exploring local file systems and executing terminal commands without host environment-specific code for each tool.

Architecture: MCP-compliant client/server pair with JSON-RPC communication over standard input/output streams

Interoperability strategy: The assistant can directly query tool capabilities using standard MCP schemas, enabling uniform treatment of file operations and database queries.

Benefits: Development is easier as you don't have to write API wrappers for each new tool.

Challenges: The agent may need a lot of filesystem access, complicating the security boundary. Running multiple tools may introduce latency.

Lesson: Open, standardised context protocols let a model discover and use new tools on its own, without custom integration code.

Case Study 3: Agent-to-Agent Communication (A2A) in Supply Chain Management

Context: Autonomous supplier, inventory, and delivery agents from different enterprises coordinate shipments in real time within a distributed logistics application.

Architecture: Decentralised multi-agent system with event-based publish/subscribe messaging over gRPC streams.

Interoperability approach: Open A2A messaging protocol for dynamic negotiation, pricing and contracts between agents.

Benefits: Less supply-chain delays, automated stock management, and fewer manual data-entry bottlenecks.

Challenges: The corporate agents can conflict when each corporate agent tries to minimise its own cost instead of the total cost of the network.

Lesson: The protocol needs to have a fallback mechanism built into it to deal with conflicts between self-interested agents.

Case Study 4: Knowledge Graphs in Enterprise AI

Context: A multinational finance corporation needs to consolidate compliance information, market analysis, and internal operational data across divisions.

Architecture: A centralised enterprise knowledge graph, queried by LLM-based agents through semantic routing.

Interoperability approach: Data are classified using W3C semantic standards (RDF/OWL), and financial concepts are mapped to precise, shared definitions consistent with the ontology-driven design-science methods described by Ben Hassen et al. (2025) for resolving fit problems in enterprise information systems.

Benefits: Fewer hallucinations, responses grounded in verified facts, and data visibility across divisions.

Challenges: Significant manual effort to keep corporate ontologies up-to-date and high computational cost at scale.

Lesson: Models based on a knowledge graph increase accuracy and verifiability.

Case Study 5: AI Interoperability in Healthcare

Context: An oncology diagnostic agent used at a research hospital must be able to analyse patient records from a separate regional clinic's database during a time-critical case.

Architecture: A decentralised healthcare system where local electronic health record (EHR) systems and bespoke diagnostic models communicate directly via microservices as opposed to a common central store.

Interoperability approach: The data exchange is arranged through Fast Healthcare Interoperability Resources (FHIR) standards combined with zero-knowledge-proof (ZKP) techniques, so that records are verifiable without exposing the underlying patient data (Topol, 2019).

Benefits: Faster clinical decisions, fewer diagnostic errors, remote expert consultation without sharing confidential records.

Challenges: How to move data fast enough to be clinically useful across organisations, whilst meeting regulatory obligations such as HIPAA.

Lesson: Medical artificial intelligence systems should be built around existing healthcare data standards such as FHIR, because privacy-preserving techniques alone cannot replace standardised data structure.

Case Study 6: Autonomous Software Engineering Agents

Context: Autonomous software-engineering agents collaborating to create a codebase.

Architecture: Multi-agent setting with different Architect, Coder, and Tester agents, each running in its respective container under a workflow manager.

Interoperability approach: Agents communicate over sockets, exchanging structured Markdown execution plans and JSON status messages.

Benefits: Less development time, automatic detection of defects by using testing, and continuous maintenance of code.

Challenges: Agents can fall into infinite loops while writing and failing tests, and long repository contexts are hard to manage.

Lesson: Independent, module-level agent work is effective, but only with strict boundary limits to prevent runaway compute costs.

Case Study 7: Multi-Agent Research Assistants

Context: Multiple agents work simultaneously to search databases, read papers, and synthesise findings for an academic-style literature review.

Architecture: A hierarchical multi-agent setup in which a manager agent assigns search vectors to lower-level retrieval agents.

Interoperability approach: A shared vector-database index and semantic search queries let agents exchange relevant article context.

Benefits: Faster synthesis, discovery of cross-field citations a single researcher might miss, and academically formatted output.

Challenges: Filtering out low-quality material from predatory journals, and reconciling contradictory findings across sources.

Lesson: Hierarchical task assignment works reliably when paired with automated data-validation steps.

Chapter 5: Research Methodology

5.1. Research Methodology

The study is based on a secondary, qualitative research design, an exploratory and descriptive approach, to explore and investigate the architectural principles and communication protocols of interoperable artificial intelligence (AI) systems. It synthesises existing academic and technical literature to surface recurring patterns, core concepts, and structural challenges associated with the AI Web.

5.2 Research Strategy

A Systematic Literature Review (SLR) approach was combined with comparative and thematic analysis. Selected studies and technical standards were compared systematically to reveal recurring architectural patterns and interoperability mechanisms across disparate AI implementations.

5.3 Data Sources

References were drawn from peer-reviewed journal articles and conference papers, together with international informatics standards and technical documentation from acknowledged organisations. Searches were run against databases and journal platforms including ScienceDirect, IEEE Xplore, ACM Digital Library, SpringerLink, Wiley Online Library, Emerald Insight, Taylor & Francis Online, and Google Scholar.

5.4 Search Strategy and Criteria

Boolean search strings used included:

("Artificial Intelligence" OR "Large Language Models") AND ("Interoperability" OR "System Architecture") AND ("Multi-Agent Systems" OR "Protocols")

("Model Context Protocol" OR "gRPC" OR "Agent-to-Agent") AND ("Distributed AI" OR "Knowledge Graphs" OR "Enterprise Information Systems")

To ensure quality and relevance, the following criteria were set for inclusion and exclusion:

Inclusion Criteria	Exclusion Criteria
Peer-reviewed journal articles and conference papers	Non-peer-reviewed blog posts or unverified opinion pieces

Studies detailing AI system architecture or protocols	Papers focused solely on training a specific neural algorithm
Literature published or updated up to 2026	Documents lacking technical implementation detail
Frameworks addressing cross-system interoperability	Studies with no clear relevance to distributed networks

Table 3. Inclusion and exclusion criteria applied during source screening.

5.5 Data Analysis Method

- Thematic analysis: identifying recurring structural themes and design concerns in the literature.
- Comparative evaluation: assessing the trade-offs of different communication protocols (REST, gRPC, MCP, A2A).
- Conceptual synthesis: integrating findings into the nine-layer AI Web architecture.

5.6 Research Workflow

The study proceeded in five steps: defining the core interoperability problem; conducting a systematic database search; screening and selecting sources against the criteria above (following PRISMA guidelines); analysing the selected material thematically and comparatively; and synthesising the findings into the architectural design presented in this paper.

Chapter 6: Findings and Discussion

6.1 Synthesis of Findings

Across the reviewed literature, achieving global AI interoperability consistently depends on moving away from proprietary, hard-coded pipelines. Enterprise, software-engineering, and multi-agent research all point to the same three needs: modular system design, open context sharing, and rigorous semantic standardisation. Without these, multi-agent systems suffer from contextual drift, in which reasoning details are lost as a task passes from one model to the next. Successful cross-platform collaboration appears to depend on decoupling system state and memory from any one underlying model, so that processing nodes can be swapped without disrupting an active workflow.

6.2 Comparison of Existing AI Architectures

Evaluating different AI deployment methods reveals distinct trade-offs regarding scalability, flexibility, and architectural complexity, summarised in Table 4.

Architecture Type	Key Strengths	Core Limitations	Interoperability Profile
Centralised AI	Fast inference; simpler data-security model.	Single point of failure; heavy vendor lock-in.	Very low; closed, proprietary environments.
Cloud-based AI	Elastic scaling; lower local hardware cost.	Network latency; recurring data-egress fees.	Moderate; usually tied to one cloud vendor.
Multi-Agent Systems	High adaptability; handles complex distributed tasks.	Heavy communication overhead; risk of infinite loops.	High within closed systems; low across platforms.
Service-Oriented Architecture	Reusable services; clear API boundaries.	Rigid schemas; high setup friction.	Moderate; depends on stable API contracts.
Emerging AI Web	Fully decentralised; flexible, ad hoc routing.	High initial complexity; needs global standards.	High; built on open, vendor-neutral protocols.

Table 4. Comparison of AI deployment architectures.

6.3 Architectural Layers Required for the AI Web

The proposed architecture separates functions into layers that do not interfere with one another. The lower layers (infrastructure and data) manage physical resources and access to data vectors. The middle layers (knowledge, model, and agent) turn raw data into usable information. The upper layers (communication, coordination, and governance) manage message passing, coordination, and security policy. This separation lets any one layer be upgraded without disturbing the rest of the network – the same design logic that Ben Hassen et al. (2025) apply at enterprise scale when resolving fit problems between business processes and information systems.

6.4 Communication Protocols Needed

The reviewed literature suggests that today's web-integration standards are not sufficient for dynamically interacting AI systems. REST APIs are too restrictive, relying on static, pre-set endpoints that do not accommodate dynamic negotiation. gRPC offers fast binary streaming well suited to microservice communication, but has no built-in semantic layer. Newer technologies such as MCP and A2A address this by standardising context discovery and intent messaging through lightweight JSON-RPC mechanisms. These protocols look set to play a role for the

emerging AI Web similar to the one HTTP and TCP/IP played for the traditional web, turning the network into a space for semantic, not just syntactic, interaction.

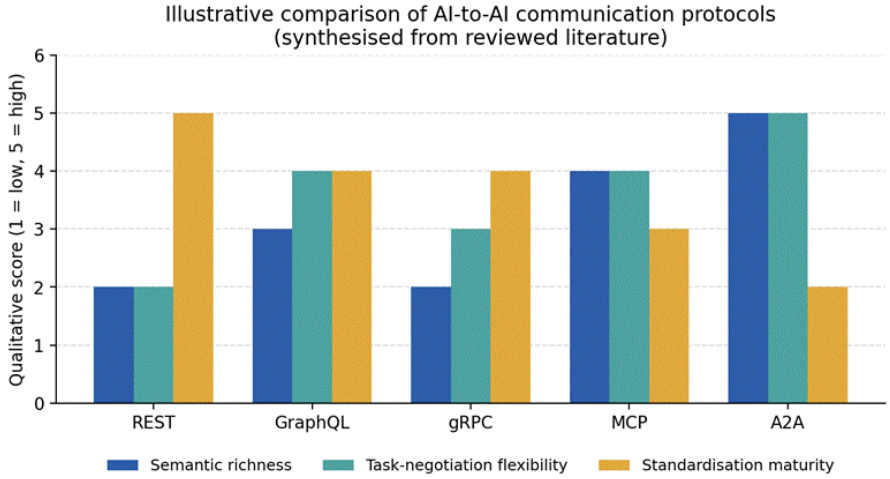

Figure 2. Illustrative comparison of five AI-to-AI communication protocols on three qualitative dimensions, synthesised from the reviewed literature.

6.5 Evaluation of Interoperability Principles

Modularity, scalability, and standardisation are consistently important when building a distributed network. Modularity keeps components separate, so a change to one part of the architecture does not ripple into every linked system. Scalability lets the network absorb changes in processing load without breaking existing connections. High explainability and auditability make it possible to review what automated agents actually did, and zero-trust models of security and privacy remain central to protecting user data as more of the network's traffic becomes agent-initiated rather than human-initiated.

6.6 Major Challenges

- Communication standards: no widely recognised, platform-neutral standard yet exists for AI-to-AI communication.
- Vendor lock-in: large technology organisations create closed ecosystems that make switching or exporting data and models difficult.
- Data privacy: exchanging data across networks and models raises serious compliance concerns under regimes such as GDPR and HIPAA.
- Cyber security: Open networks are vulnerable to prompt injection, data poisoning and unauthorised system access.
- Explainability: longer chains of agent interaction make it difficult to trace decisions and debug errors.
- Governance: The absence of clear international regulation makes it difficult to allocate responsibility when an autonomous agent malfunctions.
- AI alignment: keeping autonomous, interacting agents aligned with human intent and rules remains an open research problem.

6.7 Research Gaps

- Protocol specifications: common industry standards for AI communication beyond basic JSON-RPC structures.
- Cross-platform compatibility: translation protocols that let reasoning states move between fundamentally different architectures, such as state-space models and transformers.
- Cross-border governance: regulatory frameworks capable of overseeing decentralised, cross-border multi-agent networks.
- Benchmarking: universal tools to test interoperability, efficiency and context preservation.
- Multi-agent coordination: algorithms that resolve conflicting goals without deadlocks or infinite loops.

6.8 Implications

- Researchers should shift focus from simply scaling model parameters toward building open, collaborative communication frameworks and semantic translation tools.
- Developers should design modular systems around open standards rather than proprietary integration wrappers.
- Technology companies should invest in open, interoperable tools and agent ecosystems rather than walled gardens, to avoid losing customers to more open competitors.
- Policymakers should pursue flexible, technology-neutral regulation that enforces data-privacy mandates (such as GDPR and HIPAA) without blocking legitimate cross-platform services.

Chapter 7: Future Directions

7.1 AI-Native Internet

Digital networks appear to be heading toward an AI-native internet, in which autonomous entities, not people, are the primary producers and consumers of data. This would move the network beyond human-centric web pages and toward machine-readable semantic data streams optimised for rapid AI processing.

7.2 Internet of AI Agents

A related idea, the Internet of AI Agents, envisions a highly connected, decentralised network of specialised intelligences. Rather than interacting with large, monolithic applications, users would deploy personal agents that coordinate securely with independent corporate and logistics agents to complete complex, real-world tasks.

7.3 Standardised AI Protocol Stack

Realising this vision requires a standardised AI protocol stack, a set of rules for how each layer of communication should behave, from basic routing up through intent parsing and security validation.

7.4 Autonomous Multi-Agent Ecosystems

Such an ecosystem would comprise agents that can self-assemble and self-optimise: when a hard problem appears, agents would find each other, split the sub-tasks, share context, and verify results with minimal human input.

7.5 Federated Intelligence

Federated intelligence would allow learning and reasoning across data silos without centralising sensitive data in one place, particularly useful for international research collaborations where multiple organisations need to work with the same dataset without compromising privacy.

7.6 Decentralised AI

Decentralised ledgers and edge computing could help shield the AI Web from single points of failure and from monopolistic control by any one provider.

7.7 AI Governance and Ethical AI

As networks gain autonomy, ethical guardrails need to be built into the architecture itself. Future systems will likely need automated, cryptographically verifiable compliance tracking, so that the actions of agents truly do adhere to safety rules and legislation – a natural extension of the governance issues raised by Camilleri (2024) to a situation where the acting party is a network of agents rather than a single system.

7.8 Human-AI Interaction

The point of these architectural innovations is effective human-AI collaboration. The network should keep humans meaningfully in the loop, with transparent reasoning and control options that make it easy to supervise agents' work.

7.9 Artificial General Intelligence (AGI)

Building a secure, scalable AI Web can be seen as one stepping stone toward AGI. Connecting many narrow intelligences within one flexible, global network could give rise to system-wide problem-solving abilities that resemble general intelligence, though this remains a long-term and contested possibility rather than an established outcome.

Chapter 8: Recommendations

8.1 For Researchers

- Interoperability benchmarking: build open-source test sets that measure context preservation and semantic fidelity when tasks move between AI platforms.
- Standardised evaluation: develop metrics for the computational cost, latency, and security profile of different multi-agent coordination approaches.

8.2 For Developers

- Modular, API-based AI: break systems into microservices so that models, memory stores, and toolkits can be published and updated independently.
- Open communication protocols: adopt MCP, gRPC, and similar open standards rather than proprietary API wrappers.

8.3 For Industry

- Cross-platform collaboration: form industry partnerships to build shared, open tool ecosystems that lower integration costs and reduce vendor lock-in.
- Invest in interoperable ecosystems: shift investment from single, standalone models toward adaptive, network-based multi-agent systems.

8.4 For Policymakers

- Foster global standardisation: work with engineers on open, vendor-neutral data standards to avoid market fragmentation.
- Create adaptive regulatory frameworks: update data-privacy and liability law to account for autonomous agent activity, protecting users without slowing legitimate development.

8.5 For Standards Organisations

- Develop global AI communication standards: accelerate formal specification of machine-to-machine protocols, with strong native support for semantic translation.

8.6 For Governments

- Support governance and interoperability policy: fund public research into open, secure, privacy-preserving distributed-intelligence infrastructure, and encourage international collaboration on safety standards.

Chapter 9: Conclusion

Artificial intelligence has reached a genuine turning point: the shift from massive, isolated foundation models toward a decentralised, global network of autonomous agents, the AI Web. As this paper has argued, realising that vision is not primarily a matter of scaling model parameters further; it is a matter of building open, standardised architectures and communication protocols. By organising the system's capabilities into nine layers, and by drawing on emerging

standards such as the Model Context Protocol, developers can move beyond the constraints of the traditional REST API paradigm.

The literature reviewed here, spanning enterprise information systems, multi-agent AI, semantic web foundations, and AI governance, consistently supports the idea that modularity, loose coupling, and zero-trust security are essential to avoiding context loss and vendor lock-in. Technical challenges around data privacy, cybersecurity, and semantic drift remain serious, but they point toward a clear research agenda rather than a dead end. Building a sustainable, global AI Web will ultimately depend on a coordinated industry effort to agree on open, reliable protocols much as the early internet depended on shared agreement around TCP/IP and HTTP.

References

Aktürk, C. (2021). Artificial intelligence in enterprise resource planning systems: A bibliometric study. Journal of International Logistics and Trade, 19(2), 69–82.

Allmendinger, S., Bonenberger, L., Endres, K., Fetzer, D., Gimpel, H., & Kühl, N. (2026). Multi-agent AI. Electronic Markets, 36(1), Article 18.

Ben Hassen, M., Zahhaf, S., & Gargouri, F. (2025). Resolving the "three-fit" problems in enterprise information systems: A design science approach with ontological solutions. Business Process Management Journal, 31(6), 2518–2578.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. Scientific American, 284(5), 34–43.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., ... & Liang, P. (2021). On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258.

Bond, A. H., & Gasser, L. (Eds.). (1988). Readings in Distributed Artificial Intelligence. Morgan Kaufmann.

Camilleri, M. A. (2024). Artificial intelligence governance: Ethical considerations and implications for social responsibility. Expert Systems, 41(7), e13406.

Fareh, M. (2019). Modeling incomplete knowledge of semantic web using Bayesian networks. Applied Artificial Intelligence, 33(11), 1022–1034.

Hofmann, P., Urbach, N., Lanzl, J., & Desouza, K. C. (2024). AI-enabled information systems: Teaming up with intelligent agents in networked business. Electronic Markets, 34, Article 52.

Horvitz, E. (2022). Toward a global web of artificial intelligence: Opportunities and challenges for interconnected systems. Artificial Intelligence Review, 55(4), 2815–2839.

Hughes, L., Dwivedi, Y. K., Li, K., Appanderanda, M., Al-Bashrawi, M. A., & Chae, I. (2025). AI agents and agentic systems: Redefining global IT management. Journal of Global Information Technology Management, 28(3), 175–185.

Hughes, L., Dwivedi, Y. K., Malik, T., Shawosh, M., Albashrawi, M. A., Jeon, I., & Walton, P. (2025). AI agents and agentic systems: A multi-expert analysis. Journal of Computer Information Systems, 65(4), 489–517.

Jennings, N. R. (2001). An agent-based approach to building complex software systems. Communications of the ACM, 44(4), 35–41.

Liu, Y., Lo, S. K., Lu, Q., Zhu, L., Zhao, D., Xu, X., Harrer, S., & Whittle, J. (2025). Agent design pattern catalogue: A collection of architectural patterns for foundation model based agents. Journal of Systems and Software, 220, 112278.

Newman, S. (2021). Building Microservices: Designing Fine-Grained Systems (2nd ed.). O'Reilly Media.

Russell, S., & Norvig, P. (2020). Artificial Intelligence: A Modern Approach (4th ed.). Pearson.

Schneider, J., Meske, C., & Kuss, P. (2024). Foundation models: A new paradigm for artificial intelligence. Business & Information Systems Engineering, 66(2), 221–231.

Smith, R. G. (1980). The contract net protocol: High-level communication and control in a distributed processor node architecture. IEEE Transactions on Computers, C-29(12), 1104–1113.

Topol, E. J. (2019). High-performance medicine: The convergence of human and artificial intelligence. Nature Medicine, 25(1), 44–56.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30, 5998–6008.

Wooldridge, M. (2020). An Introduction to MultiAgent Systems (2nd ed.). John Wiley & Sons.

Zaharia, M., Chen, A., Davidson, A., Ghodsi, A., Hong, S. A., Konwinski, A., ... & Zumar, T. (2018). Accelerating the machine learning lifecycle with MLflow. Proceedings of the IEEE International Conference on Big Data, 1023–1030.