From Semantic Web and MAS to Agentic AI: A Unified Narrative of the Web of Agents

Tatiana Petrova
SEDAN - SnT
University of Luxembourg
Luxembourg, Luxembourg
tatiana.petrova@uni.lu

Boris Bliznukov
SEDAN - SnT
University of Luxembourg
Luxembourg, Luxembourg
boris.bliznukov@uni.lu

Aleksandr Puzikov
SEDAN - SnT
University of Luxembourg
Luxembourg, Luxembourg
aleksandr.puzikov@uni.lu

Radu State
SEDAN - SnT
University of Luxembourg
Luxembourg, Luxembourg
radu.state@uni.lu

***Abstract*—The concept of the Web of Agents (WoA) [1], which transforms the static, document-centric Web into a dynamic environment of autonomous agents acting on users' behalf, has attracted growing interest as large language models (LLMs) become more capable. However, research in this area is still fragmented across different communities. Contemporary surveys catalog the latest LLM-powered frameworks, while the rich histories of Multi-Agent Systems (MAS) and the Semantic Web are often treated as separate, legacy domains. This fragmentation obscures the intellectual lineage of modern systems and hinders a holistic understanding of the field's trajectory.**

In this paper, we present the first comprehensive evolutionary overview of the WoA. We show that modern protocols like Agent-to-Agent (A2A) Protocol [2] and the Model Context Protocol (MCP) [3]), are direct evolutionary responses to the well-documented limitations of earlier standards like FIPA standards [4] and OWL-based (Web Ontology Language [5]) semantic agents. To systematize this analysis, we introduce a four-axis taxonomy (semantic foundation, communication paradigm, locus of intelligence, discovery mechanism). This framework provides a unified analytical lens for comparing agent architectures across all generations, revealing a clear line of descent where others have seen a disconnect.

Our analysis identifies a fundamental paradigm shift in the 'locus of intelligence': from being encoded in external data (Semantic Web) or the platform (MAS) to being embedded within the agent's core model (LLM). This shift is foundational to modern Agentic AI, enabling the truly scalable and adaptive systems the WoA has long envisioned. We conclude that while new protocols are essential, they are insufficient for building a robust, open, and trustworthy ecosystem. Finally, we argue that the next research frontier lies in solving persistent socio-technical challenges, and we map out a new agenda focused on decentralized identity, economic models, security, and governance for the emerging Web of Agents.

***Index Terms*—Web of Agents (WoA), Agentic AI, AI Agent, Multi-Agent Systems (MAS), Large Language Models (LLM), Semantic Web, Agent Interaction Protocols, FIPA, A2A Protocol, Model Context Protocol (MCP), Agent Architectures, Functional Taxonomy, Agent Governance, Decentralized Identity**

1. Introduction

The vision of a global digital ecosystem populated by autonomous software agents, capable of discovering one another, communicating, and collaborating to perform complex tasks on behalf of users, is one of the most enduring ambitions in artificial intelligence. This concept, often termed the "Web of Agents" [1], envisions a fundamental evolution of the internet from a repository of human-readable information into a dynamic environment for machine-to-machine cooperation. Early proponents imagined distributed programs that could traverse a machine-readable web, providing novel capabilities with minimal human intervention (a "science fiction" vision that has guided research for decades). However, an analysis of the academic literature, as shown in Figure 1, reveals a nuanced history. While the specific term "Web of Agents" has seen limited traction, its foundational pillars ("Intelligent Agent" and "Multi-Agent System" (MAS)) have been cornerstones of computer science, commanding sustained and substantial research interest for over two decades. This discrepancy suggests that while the grand vision of a fully interconnected agent web remained elusive, the community has been diligently building its constituent parts. The early enthusiasm for the "Semantic Web", intended to provide the machine-readable data layer for this vision, peaked and declined, indicating that a semantically rich web was a necessary but insufficient condition for its success. Our analysis reveals that

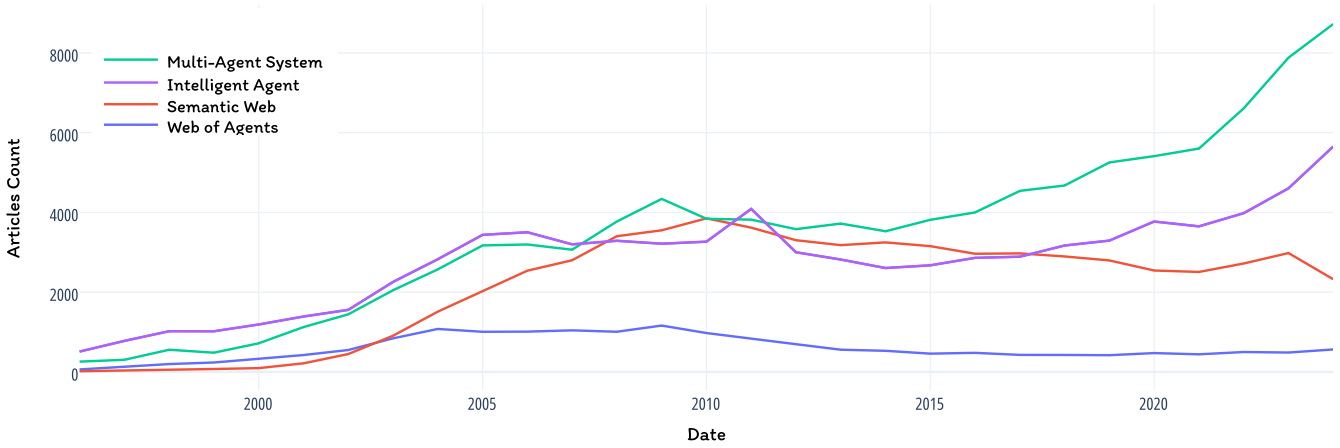

Figure 1. **Publication Trends in Agent-Related Research (1995–2024).** This graph displays the number of academic articles indexed annually in the Scopus database [14] containing key phrases related to agent technologies. The data reveals several critical intellectual currents: the rise and subsequent plateau of research in foundational fields like "Semantic Web" (amethyst) and "Multi-Agent System" (red); the persistent, low-volume discussion of the unifying "Web of Agents" (blue) concept over three decades; and the recent, exponential growth in publications on the "Intelligent Agent" (teal). This surge in practical agent technology underscores the urgent need for the robust, interoperable framework that the Web of Agents concept has long envisioned, shifting it from a theoretical construct to a practical necessity.

the primary bottleneck was not the network, but the nodes. We argue that individual agents lacked the general-purpose reasoning and adaptive capabilities to create the compelling, value-generating applications needed to drive the adoption of complex interoperability standards.

Recent breakthroughs have fundamentally altered this landscape. The advent of powerful Large Language Models (LLMs) has served as a catalyst, providing the base for a new generation of highly capable agents. Models from OpenAI (GPT-4.5 [6]), Google (Gemini 2.5 Pro [7]), Anthropic (Claude 3.7 Sonnet [8]), xAI (Grok-3 [9]), Mistral Large 2 [10]), and Alibaba (Qwen 3 [11]) demonstrate advanced capabilities in synthesizing information and generating coherent plans of action, often exhibiting emergent problem-solving skills (capabilities that are not present in smaller-scale models but arise in larger-scale models, and thus cannot be predicted by simply extrapolating the performance of their smaller predecessors [12]). Recent research posits that this apparent reasoning may stem from an "illusion of thinking," where models adeptly retrieve and adapt solution templates from their vast training data [13]. It is precisely this powerful, albeit potentially non-cognitive, mechanism that, when combined with profound language understanding, enables modern agents to interpret and manipulate web content with human-level flexibility.

Concurrently, major technology platforms have begun specifying minimal open standards to connect independent agents. For instance, Google's Agent-to-Agent (A2A) protocol aims to standardize how disparate AI agents discover and communicate with each other [2], while Anthropic's Model Context Protocol (MCP) provides a unified interface for agents to access external tools and data sources securely [3]. These developments signal the transition from isolated, platform-specific "bots" to a cohesive, interoperable Web of Agents [15], [16].

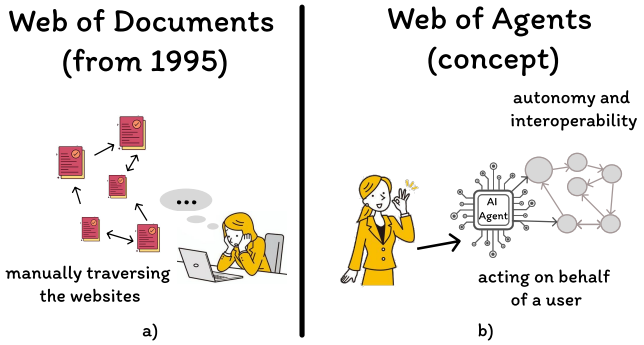

Figure 2. **The paradigm shift from the Web of Documents to the Web of Agents for solving complex tasks.** (a) **Web of Documents** (1995 – present): The user bears the full cognitive load, manually navigating static websites to find, aggregate, and process information. (b) **Web of Agents** (Concept): The user delegates a high-level goal to an autonomous AI agent. This agent offloads the cognitive load by dynamically finding and coordinating with a heterogeneous ecosystem, comprising other agents, external tools (via APIs), and data sources, to accomplish the task. This illustrates the fundamental shift from manual information integration to automated goal execution.

The idea of an intelligent, interactive web goes back to the early Semantic Web proposals in the early 2000s from Tim Berners-Lee and his colleagues [1], [17], [18], who dreamed of a "Web of Data" in which information carries well-defined meaning, so that machines and people could collaborate far more effectively. At that time, researchers envisioned personal agents capable of tasks such as automatically scheduling meetings by exchanging semantically rich information across websites [19]. Roughly at the same time, the field of Multi-Agent Systems (MAS) was emerging, studying how decentralized, autonomous entities could cooperate through negotiation and coordination, to achieve

both shared and individual goals [20].

Although Semantic Web technologies (namely RDF (Resource Description Framework) [21] and OWL (Web Ontology Language) [22]) provided a framework for agent interoperability, real-world adoption was hindered by the difficulty of large-scale ontology agreement and the high cost of semantic annotation [23]. As a result, agentic functionality remained largely confined to research prototypes and closed systems. The advent of LLMs changed this trajectory: by implicitly learning rich world representations, modern agents can bypass the need for extensive manual annotation, instead leveraging web-scale text data to reason about meaning and perform web-based actions.

To frame our analysis, we distinguish between two related but distinct concepts: **Web of Agents** is a vision of the Internet's future as a habitat for numerous interconnected agents, requiring the development of open standards and protocols for their collaboration. Meanwhile, **Agentic AI** is a conceptual framework describing the next step in AI-agent evolution: a shift toward highly autonomous multi-agent systems with collective problem-solving capabilities. While both share the philosophy of autonomous agents, Web of Agents focuses on infrastructure and integration into the Web, whereas Agentic AI emphasizes the organization and capabilities of agent teams. These approaches are complementary: WoA's ideas around standardization and interoperability can enhance Agentic AI platforms, and Agentic AI's advances in agent coordination and learning enrich the realization of the Web of Agents [24].

This survey provides a comprehensive, end-to-end analysis of the **Web of Agents**, from its historical roots to its LLM-powered future. We begin by examining the foundational work on MAS and the ambitious but ultimately flawed FIPA standards. We then analyze how LLMs have acted as a catalyst, enabling sophisticated **Agentic AI** and driving the development of new, pragmatic interoperability protocols. Finally, we explore the critical emerging frontiers of governance, security, and economics that will shape the future of this transformative technology.

1.1. Related Work and Our Contributions

While the vision of a Web of Agents has persisted for decades, its recent resurgence from a theoretical ambition to a practical engineering challenge is catalyzed by breakthroughs in large language models (LLMs). As evidenced by the long-term research focus on core agent technologies (see Figure 1), the foundational work has been extensive. LLMs have not created this field but have provided the crucial missing component: a powerful, general-purpose reasoning engine that makes individual agents highly capable and autonomous.

This rapid evolution of LLM-powered agents has spurred a number of recent survey papers. For instance, Ferrag et al. [25] provide a comprehensive review of benchmarks and frameworks for modern autonomous agents, while Yang et al. [26] offer a detailed survey of LLM-based agent architectures. Other works, such as the survey by Ehtesham et al. [27], focus specifically on comparing new agent interoperability protocols like MCP [3], A2A [2], and ANP [28].

While these recent surveys provide excellent and detailed taxonomies of modern LLM-agent frameworks and protocols, our work offers a broader historical synthesis and a unifying conceptual framework. The unique value of this paper lies not in merely cataloging the latest technologies, but in presenting a cohesive, long-term evolutionary theory of the Web of Agents. We trace the intellectual lineage of today's systems back through three decades of research in Multi-Agent Systems and the Semantic Web. We argue that the design decisions of today's lightweight, pragmatic protocols (like MCP and A2A) are a direct response to the specific challenges and limitations encountered by earlier, more formal approaches (such as the brittleness of ontologies and the complexity of FIPA).

To structure this analysis, this paper makes the following primary contributions:

- We present a unified evolutionary narrative of the Web of Agents, synthesizing three decades of research from the Semantic Web, Multi-Agent Systems, and modern LLM communities. We demonstrate how the failures and lessons of past generations have directly shaped the design principles of the current generation.
- We introduce a novel, multi-dimensional functional taxonomy for WoA architectures. Unlike existing taxonomies that focus narrowly on modern agents, our framework provides a common analytical methodology to study and compare systems across all historical generations, from FIPA-based platforms to LLM-powered agents.
- We analyze state-of-the-art technologies (including FIPA ACL, MCP, and A2A) and frameworks (such as AutoGPT and LangChain) by applying both our evolutionary narrative and functional taxonomy, clarifying their architectural trade-offs and relationships to earlier systems.
- We identify persistent challenges to achieving an open Web of Agents (interoperability, security, governance), and economic incentives. We frame them as enduring problems that each generation has attempted to solve with different technological approaches.
- Finally, based on this comprehensive historical and functional analysis, we outline promising future research directions that address the socio-technical gaps remaining for the creation of a robust, interoperable, and trustworthy multi-agent internet.

By covering these aspects, our survey serves as a comprehensive roadmap for researchers and practitioners seeking to advance the Web of Agents from isolated prototypes to a robust, interoperable multi-agent internet.

The remainder of this article is organized as follows. Section II reviews the background and traces the evolution of Web of Agents concepts. Section III introduces our novel

functional taxonomy for WoA architectures. Section IV presents the key enabling technologies and platforms in this domain. Section V surveys representative implementations and use cases. Section VI discusses the major challenges and open research issues. Finally, Section VII concludes the article and outlines future research directions.

2. Origins and Early Interpretations of the Web of Agents Concept

Today's renewed enthusiasm for the Web of Agents is driven largely by breakthroughs in large language models. However, the term itself, along with its underlying ideas, has been circulating in academic discourse for years, first appearing in research on Web Intelligence (WI) and the Semantic Web [1]. Moreover, since the expansion of the Internet was fundamentally promoted by commercial and e-commerce applications, any protocol designed for agent ecosystems must incorporate business-oriented and transactional considerations from the outset.

To understand how these early concepts evolved into today's LLM-driven agents, it is instructive to look back at the 1990s, when foundational work on intelligent software agents laid the groundwork for the Web of Agents vision. Researchers in distributed AI and multi-agent systems (MAS) developed architectures for autonomous agents that could perceive, reason, and act on users' behalf [29]. A key conceptual model was the Belief–Desire–Intention (BDI) architecture, which provided a cognitive framework for agents' decision-making based on rational plans. Early Internet agents were generally static, single-purpose programs (such as information retrieval bots, personal assistants, or price-comparison shoppers) operating in isolation without today's rich web integration. Agent communication standards began to unite: the Knowledge Query and Manipulation Language (KQML) (mid-90s) and later the FIPA Agent Communication Language (ACL) (1997–1998) defined common message formats for agents to interact [4], [30]. Crucially, in 1996 the Foundation for Intelligent Physical Agents (FIPA) was established to foster interoperability; by 1998 it had published specifications for agent communication and management that became widely adopted benchmarks. Publications in the late 90s reflected the growing optimism about web-based agents. For example, Murugesan (1998) edited *"Intelligent Agents on the Internet and Web"*, an early collection surveying how software agents could assist in web navigation, information filtering, and e-commerce tasks [31]. Around the same time, Nwana's 1996 overview of software agents categorized their capabilities and predicted their impact on the evolving World Wide Web [32]. This first generation of web agents (call it Agents 1.0) largely emphasized individual agents with fixed logic, operating on specific websites or networks in a relatively static manner.

Alongside these conceptual developments, practical agent platforms emerged by the turn of the millennium. One notable example was IBM's Aglets (first released 1997), a Java-based mobile agent toolkit that allowed programs ("aglets") to move between network hosts carrying code and state [33]. Aglets demonstrated how an agent could travel across the Web to perform tasks (e.g. fetching data from various servers), showcasing mobility and autonomy beyond the static client-server paradigm. In parallel, the MAS community produced frameworks for social agents (Agents 2.0), where multiple agents communicate and cooperate.

Alongside these conceptual developments, practical agent platforms emerged by the turn of the millennium. One notable example was IBM's Aglets (first released 1997), a Java-based mobile agent toolkit that allowed programs ("aglets") to move between network hosts carrying code and state [33]. Aglets demonstrated how an agent could travel across the Web to perform tasks (e.g., fetching data from various servers), showcasing mobility and autonomy beyond the static client-server paradigm. In parallel, the broader Multi-Agent Systems (MAS) community focused on frameworks for social agents (Agents 2.0), enabling multiple agents to communicate and cooperate through established protocols. The Java Agent DEvelopment Framework (JADE), released in 2000, provided a FIPA-compliant environment for deploying interoperable agents on a network [34]. JADE and similar toolkits enabled researchers to build agent societies that negotiated contracts, coordinated workflows, or jointly searched for information. For instance, early agent-based e-commerce experiments used communicating agents to automate buying and selling, and projects like Agentcities (2001) connected agent platforms into a worldwide testbed. By the early 2000s, the Web Intelligence research field had also united, aiming to enhance the Web with intelligent agents and mining techniques. Still, these early agent systems remained research prototypes. Integrating agents seamlessly into the open Web was difficult because AI was still in its early stages, there was no agreement on standards, and deploying agent infrastructures at Internet scale was complex.

One of the earliest documented uses of the term "Web of Agents" itself came slightly later, in Yao's 2005 discussion of Web Intelligence [35]. Yao described the Web's evolution through hierarchical levels ("Web of Data", "Web of Information", "Web of Knowledge", etc. up to the "Web of Wisdom"). He positioned the "Web of Agents" as a key facet in the future Web (alongside a "Web of Services"). In Yao's view, the Web of Agents meant "the application of intelligent agents that further amplifies the power of the Web", highlighting use cases like search agents, recommendation agents for site navigation, and automated purchasing agents [35]. Crucially, this vision was evolutionary: it emphasized extending the existing Web's capabilities with agent functionality, rather than rebuilding the Web from the ground up with a new agent-specific infrastructure. Around the same period, the annual IEEE/WIC Web Intelligence Conference (WI-IAT) and related journals firmly incorporated "Web of Agents" as a theme, indicating that by the mid-2000s the concept had entered the mainstream vocabulary of the Web Intelligence community [36], [37].

We structure the evolution of the Web of Agents into

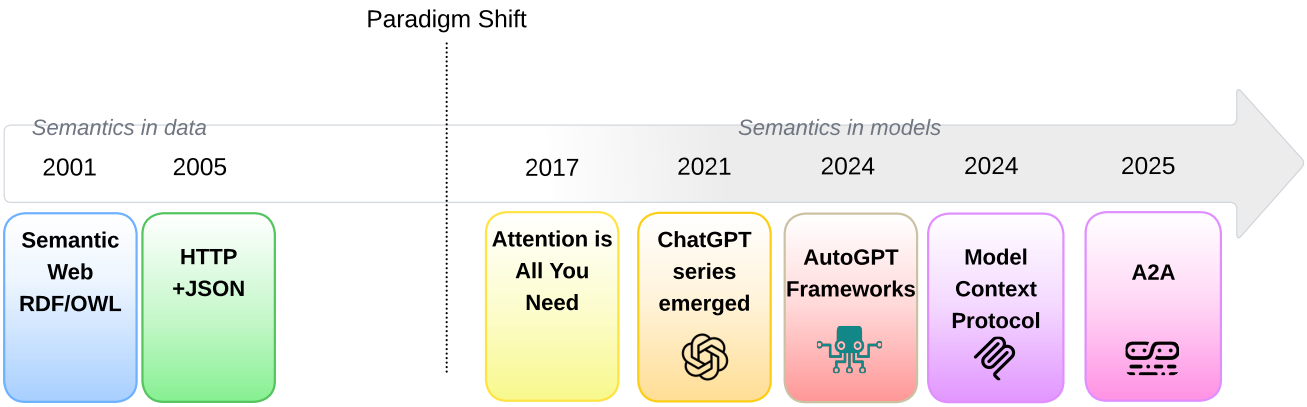

Figure 3. **The Evolutionary Paradigm Shift from 'Semantics in Data' to 'Semantics in Models'.** This timeline illustrates the fundamental shift in the "locus of intelligence" within the Web of Agents concept. Left (c. 2001-2005): **The early phase** was defined by the Semantic Web vision, where meaning was explicitly encoded in the data itself through formal, machine-readable ontologies like RDF and OWL. This was later complemented by simpler data exchange formats like HTTP+JSON. Right (c. 2017-2025): **The modern era** was catalyzed by breakthroughs in deep learning, such as the "Attention is All You Need" paper. In this paradigm, semantic understanding is implicitly embedded within the agent's core model (the LLM). The emergence of powerful LLMs like the ChatGPT series has enabled the creation of autonomous agent frameworks (e.g., AutoGPT) and lightweight, web-native interaction protocols like the Model Context Protocol (MCP) and Agent-to-Agent (A2A) protocol. This shift has decoupled agent intelligence from the need for a complex, pre-existing semantic infrastructure, facilitating the growth of more scalable and adaptable agent systems.

four successive generations to highlight the critical shifts of Web of Agents concepts. This chronology highlights how ideas matured and new technologies overcame limitations of the past:

- **Agents 1.0 (1990s).** Static, single-purpose agents with fixed logic (e.g., early bots, "softbots", email filters). Architectures such as BDI formalized agent reasoning [29], and early communication standards (KQML) provided basic interoperability. These agents operated mostly in isolation, with no large-scale web integration.
- **Agents 2.0 (2000s).** Social, communicating agents in Multi-Agent Systems (MAS). The emergence of FIPA-ACL and frameworks like JADE [34] allowed heterogeneous agents to collaborate, negotiate contracts, and coordinate workflows across networks. Examples include agent-based e-commerce negotiations and distributed problem-solving experiments.
- **Agents 3.0 (2000s–2010s).** Semantic agents leveraging the Semantic Web (RDF/OWL). These agents reasoned over machine-readable knowledge graphs, composing semantic web services and performing logic-driven inference. Envisioned scenarios included personal agents scheduling appointments or aggregating news with ontologies [1]. Adoption was hampered by the cost of ontology development and brittle inference at scale.
- **Agents 4.0 (2020s).** LLM-powered agents that learn from web-scale data. Such agents dynamically interpret natural language, browse websites, and invoke APIs via protocols like MCP/A2A. Tools like AutoGPT (2023) exemplify autonomous web navigation, task planning, and inter-agent collaboration [38]. This generation achieves a practical balance between intelligence (via LLMs) and interoperability (via lightweight web standards).

2.1. Precursors from the Semantic Web: Agents in the Machine-Readable Web

The Semantic Web initiative (introduced by Berners-Lee and colleagues) aimed to convert the document-centric Internet into a structured graph of machine-readable facts. Core standards such as RDF (Resource Description Framework), OWL (Web Ontology Language), and SPARQL (a query protocol) were introduced to encode metadata and enable automated inference across disparate sources [1], [17], [18]. Early proof-of-concepts demonstrated agents parsing RDF triples to negotiate calendar events, dynamically aligning schedules by exchanging richly annotated data between sites [19]. These ontological layers provided the vocabulary and rules that later empowered sophisticated agent ecosystems to interoperate over the Web.

2.1.1. Berners-Lee's Vision: Intelligent Agents and Semantic Understanding. The central idea of the Semantic Web was to make web data machine-readable and machine-understandable, to create a "web of data" that computers could process directly or indirectly. The Web as it existed at that time was predominantly a "web of documents" (texts and images) intended for human consumption. Computers were limited to keyword indexing and information delivery, while all the intellectual work of selecting, combining, and aggregating data fell to humans.

The Semantic Web was designed to change this state of affairs. By enriching data with clearly defined meaning, software agents could move beyond mere keyword lookup and perform true semantic searches (accounting for

synonyms, homonyms, and context), personalize websites based on users' interest profiles, and dynamically link pages according to a user's current activity. Furthermore, and crucially for the Web of Agents concept, they could aggregate information from multiple sites and carry out complex tasks automatically.

The famous article by Berners-Lee, Hendler, and Lassila (2001) [1] brought this vision to life with scenarios in which personal agents collaborate with one another and with web services to handle tasks. In their example scenario, autonomous agents coordinate to schedule a medical appointment by reasoning over distributed, machine-readable data, essentially an early Web-of-Agents narrative grounded in Semantic Web standards.

2.1.2. The Role of Ontologies (RDF, OWL) for Agent Interoperability. To implement this vision, standards were needed to represent the meaning and structure of data in a way that machines could understand. The key technologies became:

- **RDF (Resource Description Framework)**. A language for describing web resources and the relationships between them as triples (subject–predicate–object). RDF provides the basic structure for semantic data [39].
- **RDF Schema (RDFS)**. An extension of RDF that allows the definition of simple vocabularies (classes and properties) and hierarchies between them, offering basic inference capabilities (for example, property inheritance) [21].
- **OWL (Web Ontology Language)**. A more expressive language for creating formal ontologies, enabling the definition of complex classes, properties, constraints, and interrelationships between concepts. OWL is based on description logics, providing a rigorous semantics and support for automated logical inference [5], [22], [40], [41].

These technologies were designed to provide a general framework for data exchange and reuse across diverse applications, enterprises, and communities [42]. Ontologies expressed in OWL played a central role, as they allowed agents from different systems to "agree" on the meaning of terms and concepts, ensuring semantic interoperability (the ability of systems to exchange information with an unambiguous, shared understanding of its meaning) [22]. It was assumed that agents would use these ontologies to interpret data found on the Web and to communicate with one another [22].

2.1.3. Definition of Semantic Web Agents. In the context of the Semantic Web, "agents" were understood as software entities specifically designed to operate in this new, semantically enriched environment. Unlike simple web crawlers or search bots that operate at the level of syntax and keywords, Semantic Web agents were expected to possess the ability to understand the meaning of data, thanks to the presence of metadata in RDF and OWL formats [1], [18].

These agents were envisioned as "intelligent entities" capable of:

- Using ontologies to interpret information [18].
- Performing logical inference over semantic data to derive new knowledge [18].
- Interacting with semantic web services [1].
- Coordinating their actions with other agents based on a shared semantic understanding [43].
- Carrying out complex, multi-step tasks on behalf of users by integrating information from various semantically annotated sources [18], [43].

Access to the Semantic Web and its capabilities required specialized tools ("Semantic Web agents" or "semantic browsers") [1].

2.1.4. Challenges and Unrealized Potential. Despite considerable efforts to develop standards and tools, the full vision of the Semantic Web (populated by autonomous agents carrying out complex tasks) has not been realized at the scale originally envisaged. In 2006, Berners-Lee and his colleagues confessed that this "simple idea… remains largely unrealized." [44]

The reasons for this are manifold. First, creating and maintaining large-scale semantic annotations of web content proved extremely complex and labor-intensive. Consensus on ontologies was required, along with tools and incentives to annotate the vast volumes of existing and new data. Crucially, this was not merely a technical or financial hurdle, but a fundamental failure in incentive alignment [45]. The vision lacked a clear economic model to justify the immense effort required from individuals and organizations to create and maintain the semantic infrastructure [46]. Unlike the modern API economy, which is built on tangible service offerings and viable revenue generation pathways [47], the Semantic Web did not provide sufficiently compelling business cases or explicit revenue models to cover the high costs of annotation and formalization, a problem recognized early in its development [48]. This economic vacuum was a primary contributor to its limited adoption. Second, problems emerged around the scalability and efficiency of logical inference at web scale. Third, real-world web content often exhibits uncertainty, contradictions, incompleteness, and even deception, serious obstacles for systems grounded in formal logic. The AI technologies of that era weren't mature enough to deliver the autonomy and flexible reasoning needed for those ambitious scenarios. After about 2010, mainstream AI research shifted its focus toward machine learning and deep learning, and the Semantic Web receded somewhat into the background.

A central factor holding back the original vision of Semantic Web agents was not the agent technology itself, but the need for a pre-existing, sufficiently extensive, rich, and reliable semantic infrastructure (machine-readable RDF/OWL data) on which agents could depend. Without that semantic foundation, agents could not fully realize their potential to "understand" and integrate information. Traditional agent architectures have placed heavy emphasis

on formal logic and deductive inference as the foundation of agent intelligence. Modern LLM-based approaches, by contrast, rely on implicit, statistics-driven mechanisms of understanding. This shift helps explain why LLM-powered systems are often considered more pragmatic for real-world web scenarios involving imperfectly structured data.

2.1.5. Contemporary Realizations of the Semantic Web Vision. While the full Semantic Web hasn't been realized, its influence is evident in several modern web practices that embed machine-readable semantics into content. For instance, Facebook's Open Graph protocol (an RDFa-based metadata schema) [49] allows webpages to publish rich semantic descriptors (author, title, type, etc.), reflecting exactly the kind of page-annotation originally envisioned for the Semantic Web [50]. Likewise, Wikipedia's structured infobox data (made available via projects like DBpedia [51] and Wikidata [52]) feeds into large knowledge bases. Google's own Knowledge Graph (displayed in knowledge panel infoboxes) draws heavily on such semantically-structured data to present integrated facts to users [53]. More broadly, the adoption of semantic markup standards RDFa, Microdata, and JSON-LD through the Schema.org initiative has become widespread across the Web, enabling site owners to embed interoperable descriptions of people, places, events, and things into their HTML pages [54]. These practices are widely recognized as partial fulfillments of the Semantic Web vision. They implement machine-readable, shared vocabularies in web content and power intelligent applications (from richer search results to personal digital assistants), albeit in domain-specific or centralized ways rather than as a single universal graph. In this way, important parts of Berners-Lee's original vision (a web of data that machines can understand and reuse) are appearing in today's online systems, although in a fragmented, practical form.

2.2. Multi-Agent Systems (MAS) and the Web: Parallel Visions and Interoperability Challenges

Parallel to the rise of the Semantic Web, the field of Multi-Agent Systems (MAS) was developing its own approaches to building systems out of interacting autonomous entities [40], [55]. Although the goals of MAS and the Semantic Web overlapped (both aimed to create decentralized, open systems in which components could collaborate to tackle complex tasks), their methodologies diverged, leading to integration challenges, for example around differing protocols and standards for agent interaction [4].

2.2.1. Core Principles of MAS. Research in Multi-Agent Systems (MAS), which dates back to the Distributed Artificial Intelligence (Distributed AI, DAI) movement of the late 1970s [56], focuses on designing and analyzing systems made up of multiple agents. In this context, an agent is an autonomous entity (software or physical) capable of perceiving its environment, making decisions, and acting to achieve its goals [56].

The key characteristics of MAS are:

- **Autonomy**. Agents operate without direct external control, managing their own state and behavior [56].
- **Decentralization**. There is no single control center; decision-making and control are distributed among the agents [57].
- **Local Perception and Control**. Agents typically have access only to local information and can affect only their immediate surroundings [58].
- **Interaction**. Agents communicate and coordinate their actions to tackle tasks that cannot be solved individually [56].
- **Social Behavior**. Agents may follow norms, assume roles within organizations, and form social structures [56].

Early work in DAI/MAS concentrated on applications such as distributed sensor monitoring (e.g., the Distributed Hearsay-II project [59]) and distributed planning and coordination (e.g., Partial Global Planning within the Distributed Vehicle Monitoring Testbed framework [60]).

2.2.2. FIPA Standards and Agent Communication Languages (ACL). To enable interaction among agents developed by different teams on varied platforms, a need for standardization emerged. The Foundation for Intelligent Physical Agents (FIPA), founded in 1996, played a key role in this effort [61]. FIPA's objective was to develop widely accepted specifications for agent technologies, thereby fostering the creation of interoperable and open MAS [62].

The centerpiece of FIPA's standards is the Agent Communication Language (FIPA ACL), a language designed for agent-to-agent interaction [63]. FIPA ACL is grounded in speech-act theory, where each message represents a specific communicative act (or performative) that conveys the sender's intention (such as `request` (ask for an action), `inform` (report a fact), `agree` (accept a request), `refuse` (decline), or `query-ref` (request information)). A FIPA ACL message follows a standardized structure, including fields for specifying the sender and recipient(s), the performative itself, the content language (`:language`), the ontology in use (`:ontology`), the interaction protocol (`:protocol`), and the message's content (`:content`) [64].

An important distinction of FIPA ACL from its predecessor, KQML (Knowledge Query and Manipulation Language), was the inclusion of a formal semantics grounded in the modal logic of beliefs, desires, and intentions (BDI logic) [4], [30]. This semantics specifies the preconditions and expected rational effects of each communicative act, ensuring agents can interpret messages more precisely and unambiguously.

FIPA also standardized the architecture of the agent platform—defining components such as the Agent Management Service, the Directory Facilitator for discovering agents and services, and the Message Transport Service (as well as a set of Interaction Protocols, e.g., the FIPA Request Interaction Protocol, the FIPA Query Interaction Protocol, and the FIPA Contract Net Interaction Protocol) [4].

2.2.3. Bridging the Gap: Integrating MAS with Web Technologies. Despite shared goals of decentralization and openness, integrating MAS built on FIPA standards with the Web's architecture has proven to be a significant challenge [23]. The Web architecture (particularly in its RESTful interpretation) leans toward simplicity, stateless interactions, and resource orientation via URIs [65]. MAS interaction models, by contrast, often require more complex, semantically rich, and stateful dialogues between agents, driven by FIPA ACL and shared ontologies [23], [65].

Various efforts have been made to bridge these two worlds:

- **Using web technologies to represent agents.** [66] Applying RDF and Linked Data to describe agents and their services, making them accessible within the Web's hypermedia environment.
- **Leveraging Semantic Web languages in MAS.** [67] Employing OWL to define the ontologies that agents use in their FIPA ACL messages to describe content, or even to model the agents' own knowledge and beliefs.
- **Adapting MAS governance models for the Web.** [68] Attempts to implement MAS governance concepts (norms, institutions, policies) using web components and protocols.

However, these approaches have struggled to map the abstract and often complex MAS models onto the Web's more concrete, constrained components, all while preserving core web principles such as scalability, loose coupling, and simplicity.

2.2.4. Middleware-Based Approaches. One pragmatic solution for overcoming the incompatibility between FIPA-compliant MAS and standard web services (initially often based on SOAP/WSDL/UDDI) was to employ middleware.

A prominent example of this approach is the Agent Web Gateway, proposed by Shafiq et al. [69]. The primary goal of the Agent Web Gateway was to provide dynamic and seamless interoperability between MAS and Web services without requiring changes to existing FIPA or W3C Web service specifications. The gateway acted as a bi-directional translator, performing the following functions [69]:

- **Service discovery translation.** Converting service discovery requests between the FIPA Directory Facilitator (DF) and a Web service registry (e.g., UDDI).
- **Service description translation.** Transforming service descriptions between MAS formats (e.g., FIPA SL-based ontologies) and Web service formats (e.g., WSDL, possibly extended with semantic annotations such as OWL-S or WSMO).
- **Communication protocol translation.** Mapping FIPA ACL messages to Web service invocations (e.g., SOAP requests) and vice versa.

Similar gateways, such as the Web Service Integration Gateway (WSIG), were also implemented within popular agent platforms like JADE [70]. These solutions enabled FIPA agents to discover and invoke standard Web services, while Web clients could interact with services provided by agents [71].

2.3. Related Surveys

The vision of a "Web of Agents" (WoA) builds upon several established research threads, including agentic AI, multi-agent systems, and Semantic Web technologies. Recent publications have begun to survey these areas individually. For example, *Agentic AI*, referring to autonomous AI agents that pursue complex goals with minimal human intervention, has been the focus of a comprehensive survey in IEEE Access (2025) that delineates its core concepts, methodologies, and applications [72]. This survey defines the distinguishing characteristics of agentic AI (e.g. adaptability, self-directed tool use) and discusses its ethical and governance challenges, serving as a foundational reference on the emerging paradigm. Schneider (2025) provides a complementary perspective by contrasting generative AI and agentic AI, elucidating how agentic systems extend beyond static generative models through stronger reasoning and interactive capabilities [73]. Meanwhile, Sharma *et al.* (2025) argue in a recent position paper for an open "Web of Agents" architecture to prevent fragmented agent ecosystems [74]. They propose a minimal interoperable framework, comprising standards for agent communication, discovery, and state management, built on existing Web protocols. These works collectively establish the context for agentic systems on the Web. However, they stop short of providing a unifying survey of WoA that integrates insights from classical multi-agent research and Web semantics. Our survey addresses this gap by combining these threads and focusing on the interoperability of agents at web scale.

The field of *Multi-Agent Systems* (MAS) offers decades of foundational work on collections of autonomous agents. An early landmark is the "roadmap" article in the inaugural issue of *Autonomous Agents and Multi-Agent Systems* (1998), which surveyed fundamental MAS themes such as agent architectures, communication, cooperation, and negotiation strategies [75]. This work charted the research agenda and challenges (e.g. coordination and rationality principles) that shaped the MAS community. More recently, Dorri *et al.* (2018) published an extensive MAS survey in IEEE Access covering definitions, features, applications (from smart grids to network routing), and open challenges in the domain [76]. It offers a taxonomy of MAS approaches and evaluates issues like security, scalability, and agent communication protocols. These surveys provide broad coverage of MAS in traditional settings (often assuming pre-programmed agents in controlled environments). In contrast, the present work extends this line of inquiry to modern agentic AI contexts and web-enabled agents. We examine how the principles identified by earlier MAS surveys (such as decentralized problem solving and coordination mechanisms) apply when agents are powered by large language models and connected through Web standards. By doing so, our survey bridges

classic MAS research with contemporary "agentic" systems, highlighting how WoA differs from prior multi-agent frameworks (e.g. greater use of learning-based autonomy and natural language coordination).

Another pillar of WoA is the progress in *Semantic Web technologies* that enable machine-readable knowledge on the Web. The Semantic Web vision was famously introduced by Berners-Lee and colleagues in 2001 (Scientific American), which painted a scenario where software agents leverage ontologies and metadata to perform complex web tasks for users [1]. This foundational article established key standards (RDF, OWL) and envisioned an ecosystem of interoperating agents "roaming" the Web to execute sophisticated queries and services. Five years later, an IEEE Intelligent Systems article revisited the state of the Semantic Web, noting incremental adoption of standards alongside ongoing challenges in reasoning, data integration, and trust [77]. That retrospective underscored that, while ontologies and linked data were maturing, the agent-oriented aspects of the Semantic Web had yet to fully materialize. In the two decades since, Semantic Web research has produced robust knowledge representation frameworks and query languages (SPARQL, etc.), which are highly relevant to today's agentic systems. However, prior surveys of Semantic Web and Linked Data (e.g. [77]) generally do not focus on autonomous agents as first-class actors, treating them abstractly or as future applications. Our work differentiates itself by examining how modern AI agents can directly exploit Semantic Web infrastructure for discovery, communication, and knowledge sharing in an open environment.

We highlight how WoA can leverage ontologies and linked data to achieve interoperability across diverse agent platforms. A topic that lies at the intersection of semantic technologies and MAS but has not been comprehensively surveyed to date. In summary, whereas earlier surveys have studied agentic AI, multi-agent systems, and Semantic Web technologies mostly in isolation, our survey provides an integrated overview that spans all three domains. We build on the scope and insights of prior work. For instance, adopting the terminology and challenges identified by MAS surveys [75], [76] and the standards and vocabularies from Semantic Web literature [1], [77]. But we extend these discussions to the novel setting of a Web of Agents. By comparing and synthesizing these lines of research, we shed light on how the WoA paradigm both differs from and complements earlier frameworks. In particular, our survey emphasizes how interoperability standards (rooted in the Semantic Web) combined with the autonomy and learning capabilities of agentic AI can overcome the isolation characteristic of recent agent ecosystems [74]. This approach allows us to outline a research landscape that bridges classic MAS theory with modern web-based agents, positioning our work as a unifying update to the literature and a foundation for further developments in the Web of Agents era.

2.3.1. Limitations of Early Integration Efforts. Despite the existence of FIPA standards and the development of middleware solutions [62], [78], interoperable MAS never gained widespread adoption on the open Web [23]. Experiments such as Agentcities demonstrated that even when adhering to FIPA standards, achieving true interoperability between independently developed agents required additional agreements and clarifications beyond the base specifications [23].

Possible reasons for the limited success include:

- **FIPA complexity.** The formal semantics and rich set of communicative acts in FIPA ACL, though powerful, could be excessively complex for many web applications and difficult to implement fully and correctly [62].
- **Architectural mismatch.** The fundamental differences between MAS interaction models (dialogue- and state-oriented) and the Web (resource-oriented and stateless) created a persistent tension that middleware could only mask, not resolve [79].
- **Semantic interoperability challenges.** Even with a shared syntax (FIPA ACL) and specified ontologies, achieving genuine mutual understanding between agents remained difficult due to divergent interpretations of ontologies or incomplete specifications [64].
- **Lack of "killer" applications.** There may have been insufficiently compelling or scalable use cases to justify the complexity of deploying and maintaining FIPA-compliant MAS in a web environment [23].

The history of MAS–Web integration reveals a constant tension between the rich yet complex MAS models and the pragmatic but simpler Web architecture. Standardization efforts (FIPA) were necessary but not sufficient [69], [71]. Practical issues of discovery, semantic alignment, and integration with Web infrastructure led to the creation of middleware solutions [69], [71], and eventually to the search for new approaches (seen today in protocols like MCP and A2A [2], [3]) that more tightly integrate with core Web standards. The evolution from FIPA ACL to the Agent Web Gateway and onward to MCP/A2A reflects the trend of leveraging existing, widely adopted Web standards (HTTP, JSON-RPC) instead of building separate, agent-specific communication stacks [2], [3], [69].

This historical analysis reveals that past approaches can be distinguished by fundamental architectural choices regarding how they establish meaning, communicate, and locate intelligence. To systematize this comparison and provide a robust framework for analyzing both past and future systems, we now introduce a multi-dimensional functional taxonomy designed specifically for Web of Agents architectures.

3. A Taxonomy of Web of Agents Architectures

To move beyond a chronological review and provide a more robust analytical framework, we introduce a functional taxonomy of Web of Agents (WoA) architectures. While other taxonomies for multi-agent systems exist, often focusing on agent properties or environmental characteristics [80], our taxonomy is specifically designed to classify and

Table 1. APPLICATION OF THE FUNCTIONAL TAXONOMY TO WOA GENERATIONS

Taxonomic Dimension	FIPA-based MAS	Semantic Web Agents	Modern LLM-based Agents
Semantic Foundation	**Procedural Semantics:** Meaning is derived from the formal semantics of communicative acts in FIPA ACL.	**Formal/Explicit Semantics:** Meaning is encoded in external, shared ontologies (RDF, OWL).	**Implicit/Emergent Semantics:** Meaning is inferred by the LLM's internal world model from natural language descriptions.
Communication Paradigm	**Performative-based:** Uses rich speech-act theory (FIPA ACL).	**Resource-Oriented / Procedural:** Interacts with semantic web services or uses ACLs for communication.	**RPC / Resource-Oriented:** Primarily uses lightweight RPC (MCP, A2A) and RESTful APIs for tool interaction.
Locus of Intelligence	**Intelligence-in-Platform:** The FIPA platform (Directory Facilitator, etc.) provides core coordination services.	**Intelligence-in-Data:** Agent intelligence relies on reasoning over richly annotated semantic data.	**Intelligence-in-Agent/Model:** The LLM itself is the primary locus of reasoning, planning, and understanding.
Discovery Mechanism	**Centralized Registry:** Relies on a platform-specific Directory Facilitator (DF) to find other agents.	**Centralized Registry / Querying:** Depends on querying known semantic repositories or registries.	**Standardized Metadata File / Decentralized:** Moves towards decentralized discovery via well-known files (A2A's `agent.json`) or future P2P networks.

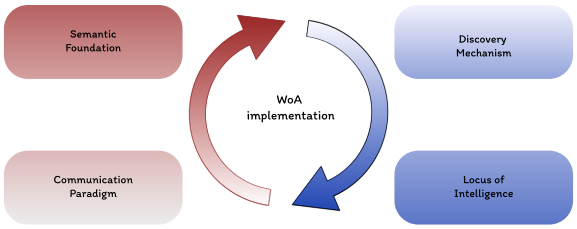

Figure 4. **The Dimensions of the Functional Taxonomy for Web of Agents Architectures.** The figure presents a multi-dimensional taxonomy designed to classify and compare different generations of Web of Agents (WoA) architectures. This framework organizes systems based on how they address the fundamental challenges of interoperability: achieving mutual understanding, exchanging messages, locating intelligence, and discovering peers. The taxonomy serves as a unified analytical framework for comparing all generations of systems, from FIPA-based platforms to LLM-powered agents. The four key dimensions are: Semantic Foundation defines how agents establish a shared understanding of the concepts, tasks, and data they interact with; Communication Paradigm classifies the primary style or protocol used for message exchange between agents or between an agent and its tools; Locus of Intelligence identifies where the core reasoning, planning, and decision-making capabilities of the system reside; Discovery Mechanism defines how agents find each other and learn about their respective capabilities in a distributed environment.

compare the architectural principles underpinning different generations of WoA. This multi-dimensional framework organizes systems based on how they address fundamental challenges of interoperability: achieving mutual understanding, exchanging messages, locating intelligence, and discovering peers. This taxonomy will serve as a consistent reference point throughout the remainder of this survey, allowing for a structured comparison of systems ranging from early Semantic Web concepts to modern LLM-based agent frameworks.

3.1. Semantic Foundation

The semantic foundation describes how agents establish a shared understanding of the concepts, tasks, and data they interact with. This is the cornerstone of interoperability. The application of the Functional Taxonomy to WoA Generations is presented in Table 1.

- **Formal/Explicit Semantics.** This approach relies on predefined, machine-readable ontologies and formal knowledge representations. Agents achieve mutual understanding by committing to a shared, explicit model of the world, typically expressed in languages like the Resource Description Framework (RDF) and the Web Ontology Language (OWL). This was the foundational vision of the Semantic Web, where data itself was imbued with meaning.
- **Procedural Semantics.** In this model, meaning is derived from the execution of standardized communication protocols and interaction patterns. Understanding is achieved not through a shared world model, but through adherence to the formal semantics of communicative acts defined in an Agent Communication Language (ACL) [4]. The FIPA ACL, for instance, defines the preconditions and rational effects of performatives like `request` or `inform`, allowing agents to reason about the intent of a message based on the protocol itself [62].
- **Implicit/Emergent Semantics.** This modern approach leverages the reasoning and natural language understanding capabilities of Large Language Models (LLMs). Shared understanding is not explicitly programmed via ontologies or protocols but emerges from the LLM's ability to interpret descriptions, instructions, and context provided in natural language. The semantics are "in the model," which can infer meaning from unstructured text and API documentation without requiring a formal, shared knowledge base.

3.2. Communication Paradigm

This dimension classifies the primary style or protocol used for message exchange between agents or between an agent and its tools.

- **Performative-based (Speech Act Theory).** Communication is modeled on human speech acts, where each message is a "performative" that conveys a specific intent. For example, a message is not just data but an explicit `inform`, `query`, or `propose` action. This paradigm, exemplified by FIPA ACL, enables rich, stateful dialogues and complex negotiations [4].
- **Remote Procedure Call (RPC).** Agents interact by invoking methods or functions on one another. This is an action-oriented paradigm where one agent requests another to execute a specific procedure with a given set of parameters. The lightweight JSON-RPC protocol, used by modern standards like MCP and A2A, is a prime example of this approach, prioritizing simplicity and web-native integration [27].
- **Resource-Oriented (RESTful)**. Agents interact by manipulating representations of resources through a uniform interface, typically using standard HTTP methods (GET, POST, PUT, DELETE). While not a pure agent-to-agent protocol, this is a dominant paradigm for how agents interact with the vast majority of existing web services and APIs, often through middleware or dedicated tool integrations.

3.3. Locus of Intelligence

This dimension identifies where the core reasoning, planning, and decision-making capabilities of the system reside.

- **Intelligence-in-Data.** In this architecture, the agents themselves are relatively simple, acting as processors of richly structured, semantically annotated data. The "intelligence" is encoded in the knowledge graphs (e.g., RDF/OWL) that the agents consume. The primary task of the agent is logical inference over this pre-existing, machine-readable data, as envisioned in the early Semantic Web.
- **Intelligence-in-Platform/Middleware.** Here, the agent platform provides the bulk of the intelligence and coordination services. The platform manages agent discovery (e.g., via a Directory Facilitator), enforces complex interaction protocols, and handles lifecycle management. Individual agents plug into this sophisticated infrastructure, which orchestrates their behavior. The FIPA/JADE architecture is the classic example of this model.
- **Intelligence-in-Agent/Model.** The agent itself, powered by a sophisticated core model like an LLM, is the primary locus of intelligence. The agent possesses the intrinsic capability to understand high-level goals, reason, plan, and interact with simple, unstructured interfaces (like natural language API descriptions). The surrounding infrastructure is lightweight, providing connectivity rather than cognitive services. This is the dominant paradigm in modern agentic systems [81].

3.4. Discovery Mechanism

Discovery mechanisms define how agents find each other and learn about their respective capabilities in a distributed environment.

- **Centralized Registry.** Agents discover each other by querying a dedicated, well-known service that acts as a "yellow pages". In FIPA-compliant systems, this role is filled by the Directory Facilitator (DF). In the era of Web Services, the Universal Description, Discovery, and Integration (UDDI) registry served a similar purpose.
- **Standardized Metadata File.** An agent advertises its capabilities by hosting a machine-readable file at a standardized, well-known location. A discovering agent can fetch and parse this file to learn how to interact with the target agent. The "Agent Card" (`agent.json`) proposed in the A2A protocol is a contemporary example of this decentralized, file-based discovery approach [2].
- **Decentralized/Networked.** Discovery occurs through peer-to-peer mechanisms without reliance on a central authority. This can range from simple network broadcasts to more advanced proposals like the Agent Network Protocol (ANP), which envisions using decentralized identifiers (DIDs) and peer-to-peer networks to create a fully decentralized discovery and trust fabric [28].

By applying these four dimensions, we can precisely classify any given WoA implementation (be it a JADE-based multi-agent system, a Semantic Web agent, or an AutoGPT-style autonomous agent). And create a clear, comparative map of the field. This framework not only helps in understanding the historical evolution but also in highlighting the fundamental architectural trade-offs that continue to shape the future of the Web of Agents.

3.5. Planning Paradigms and Their Architectural Implications

While the preceding taxonomy defines the high-level architectural blueprint of a WoA system, an agent's ability to plan is not independent of this design. The choice of a planning paradigm is deeply intertwined with the architectural dimensions, particularly the Locus of Intelligence and Semantic Foundation. For example, an architecture where intelligence resides in the data and relies on formal semantics (e.g., the Semantic Web vision) is a natural fit for classical, logic-based planners that require structured domain models. In contrast, the modern architecture where intelligence is located in the agent/model and leverages implicit semantics

is precisely what enables the more flexible, LLM-native planning frameworks. Therefore, to fully appreciate the functional consequences of these architectural choices, it is crucial to examine the planning methodologies they enable. This section analyzes the three primary families of planning algorithms (classical/hierarchical, probabilistic, and LLM-native) and clarifies their connection to the broader WoA architectures.

Classical and Hierarchical Planning. Classical planning, with its formal languages like the Planning Domain Definition Language (PDDL), offers structured and verifiable plan generation. In PDDL, a planning problem is defined by a domain (actions and their preconditions/effects) and a problem instance (objects, initial state, and goal state). The primary strength of these planners is their ability to guarantee the logical correctness of a generated plan, assuming the model is accurate. However, creating these formal models for the complex and unstructured web is a significant bottleneck.

To bridge this gap, recent research integrates the semantic understanding of LLMs with the formal rigor of classical planners. A notable example is the TWOSTEP framework [82], where an LLM first decomposes a high-level, multi-agent task into sub-tasks and then translates them into a PDDL-compatible format. This allows a classical planner to solve for a concrete, low-level plan, effectively combining the strengths of both paradigms.

Hierarchical Task Networks (HTNs) offer a more natural approach for complex, multi-step web tasks. Instead of searching a state space, HTN planners decompose abstract tasks into smaller, more concrete sub-tasks based on predefined methods or "recipes" [83]. This hierarchical structure mirrors human-like problem-solving and is well-suited for web automation tasks like booking a flight, which involves a known sequence of steps (e.g., search, select, fill passenger details, pay).

However, such structured approaches fundamentally rely on a predictable world model where actions have deterministic outcomes. This assumption is often violated in the dynamic web environment, where an API call might fail, a website's layout can change, or the agent's perception of the state is incomplete. To operate robustly under these conditions, agents require mechanisms to reason about and plan under uncertainty.

Probabilistic Planning. The web is inherently stochastic; actions can fail, and the agent's perception of the environment is often incomplete or noisy. Probabilistic planning models are designed to handle such uncertainty. The primary frameworks used are Markov Decision Processes (MDPs) and their more general extension, Partially Observable Markov Decision Processes (POMDPs).

A POMDP models the world through states, actions, and observations, along with probabilistic transition and observation functions. Solving a POMDP yields a policy that maps belief states (probability distributions over states) to actions, allowing the agent to act optimally under uncertainty. Integrating POMDPs into agent architectures, such as the Belief-Desire-Intention (BDI) framework AgentSpeak+, has been shown to be effective [84]. In this integration, the agent's belief base is extended to handle probabilistic beliefs, enabling it to reason about and plan for uncertain outcomes during web interactions.

LLM-native Planning Frameworks. The advent of powerful LLMs has spurred the development of novel planning frameworks where the LLM itself is the core planner. These methods leverage the model's vast world knowledge and reasoning capabilities. An early and influential approach is ReAct (Reason and Act) [81], which demonstrates that LLMs perform better when they explicitly generate reasoning traces and interleave them with actions (e.g., tool use). The reasoning trace serves as a form of dynamic, lightweight plan.

This concept has been extended to more sophisticated structures. The Tree of Thoughts (ToT) framework [85] moves beyond the linear, chain-of-thought reasoning of ReAct. It enables the LLM to explore multiple reasoning paths simultaneously in a tree structure, allowing for self-evaluation and backtracking, which leads to more deliberate and robust problem-solving. Building on this, ReAcTree [81] uses a dynamic tree structure for hierarchical task planning, where nodes in the tree are themselves LLM agents, leading to improved task decomposition and execution.

A more recent trend is the explicit separation of high-level planning from low-level execution. Frameworks like Plan-and-Act [86] employ two distinct models: a Planner LLM that formulates a high-level plan and an Executor LLM that carries out the next step based on the plan and current environmental feedback. This architecture allows for dynamic replanning at each step; if the executor fails or encounters an unexpected state, the planner can be reinvoked to create a new plan, making the agent more adaptive.

Comparative Analysis and Future Directions. Each planning paradigm presents a unique set of trade-offs, summarized in Table 2. Classical and hierarchical planners provide high verifiability but struggle with the unstructured nature of the web and require significant domain engineering. Probabilistic planners are theoretically robust in handling uncertainty but face high computational complexity. LLM-native frameworks offer unparalleled flexibility and require no formal domain modeling but can be less reliable and their reasoning process is often opaque.

Future research is likely to focus on hybrid approaches that synergize these methods. For instance, using LLMs for high-level, common-sense planning and goal formulation, while employing more formal planners for critical, low-level execution steps where safety and correctness are paramount. Furthermore, developing methods for lifelong learning of planning strategies and improving the efficiency and verifiability of LLM-based planners are key open challenges.

4. The Modern Agent Stack: From Cloud Orchestration to Interaction Protocols

The past few years have witnessed a new surge of interest in the concept of the Web of Agents, catalyzed

Table 2. COMPARISON OF PLANNING PARADIGMS FOR WEB AGENTS

Paradigm	Strengths	Weaknesses	Example(s)
Classical / HTN	Verifiable plans, Structured decomposition, High precision	Requires formal model, Brittle to unexpected changes	PDDL, HATP [83], TWOSTEP [82]
Probabilistic	Principled uncertainty handling, Optimal policies under noise	High computational complexity, Requires probabilistic model	POMDPs in AgentSpeak+ [84]
LLM-native	High flexibility, Zero-shot generalization, No formal model required	Potential for hallucination, Less verifiable, High inference cost	ReAct [81], ToT [85], Plan-and-Act [86]

by breakthroughs in large language models (LLMs) [87], [88] and the realization of the need for new standards for AI-agent interaction in the Web environment [3], [89]. A number of key enabling technologies have now converged to make this vision feasible. These include:

1) The rise of large-scale LLMs and advanced reasoning models that imbue agents with unprecedented natural language understanding and planning abilities [81], [88], [90].
2) Improved means for agents to interface with the Web, such as web scraping tools, DOM/XPath parsing of HTML, direct API integrations, and even browser automation via frameworks like Selenium—allowing agents to navigate and extract information from online resources.
3) New training and adaptation methods that enhance agent performance, including reinforcement learning (RL) for long-horizon decision optimization, imitation learning from human demonstrations, and retrieval-augmented generation (RAG) for dynamic knowledge access. RAG is a technique that enhances language models by allowing them to retrieve information from an external knowledge base. This retrieval process helps ground the model's responses in factual data, reducing hallucinations and enabling access to up-to-date or domain-specific knowledge. The retrieved information is then used as context to generate more accurate and informed outputs [91].
4) Semantic technologies (RDF, OWL, and knowledge graphs) that provide structured representations of knowledge and can be leveraged alongside unstructured data [1], [19].
5) Powerful infrastructure (cloud computing services, containerization and Kubernetes orchestration, distributed computing frameworks) enabling scalable deployment of agent ecosystems.
6) Emerging solutions for an agent's memory management and tool-use. E.g., vector-database memories and plugin/tool ecosystems (such as OpenAI's ChatGPT plugins [92] and libraries like LangChain [93], [94]) that extend an agent's capabilities through standardized interfaces.

Notably, early attempts at agent communication standards (the Knowledge Query and Manipulation Language (KQML) [30] and the FIPA Agent Communication Language (FIPA ACL) [63], [64]) laid a foundation by defining structured message protocols with formal semantics, but these were oriented toward controlled environments and did not fully anticipate the scale and heterogeneity of the open Web. This historical context is crucial, as the design philosophy of modern protocols represents a direct reaction to the limitations of these earlier, more complex systems. The failure of heavyweight, specialized stacks like FIPA to gain widespread adoption taught a critical lesson: pragmatic simplicity and alignment with existing web standards are paramount for success. This shift represents a direct response to the lessons of the past. Reacting to the complexity of FIPA, modern protocols like MCP and A2A consciously trade of lightweight, RPC-style interactions built on ubiquitous technologies like HTTP and JSON.

4.1. Deployment Architectures: From Borg to Kubernetes

Effective deployment and management of complex agentic systems at scale require robust orchestration platforms. While containerization (e.g., using Docker) solves the problem of dependency isolation, managing the lifecycle of thousands of containers across clusters of hundreds of machines requires orchestration systems. Two prominent examples of such systems are Borg, Google's internal system, and its open-source successor, Kubernetes [95].

Kubernetes has become the industry standard for deploying cloud-native applications, including AI and machine learning workloads [96]. Its architecture provides high scalability, fault tolerance, and portability. A key innovation in Kubernetes is the Pod, an atomic deployment unit that can contain one or more tightly coupled containers sharing common resources, such as a network space. This simplifies inter-component communication for agents [97].

Borg was created by Google to manage its massive clusters and served as the blueprint for Kubernetes. It operated on the concept of jobs, which were executed as a set of tasks within containers. Borg demonstrated the effectiveness of large-scale orchestration, managing hundreds of thousands of jobs across tens of thousands of machines. However, its architecture was monolithic and tightly integrated with Google's infrastructure [98].

Kubernetes inherited key principles from Borg but introduced a more flexible and universal model. The main lessons learned from a decade of operating Borg led to improvements in Kubernetes, such as a labeling system for grouping resources, a more advanced networking model (IP-per-Pod), and a declarative API, making it a powerful tool for managing distributed systems in any environment

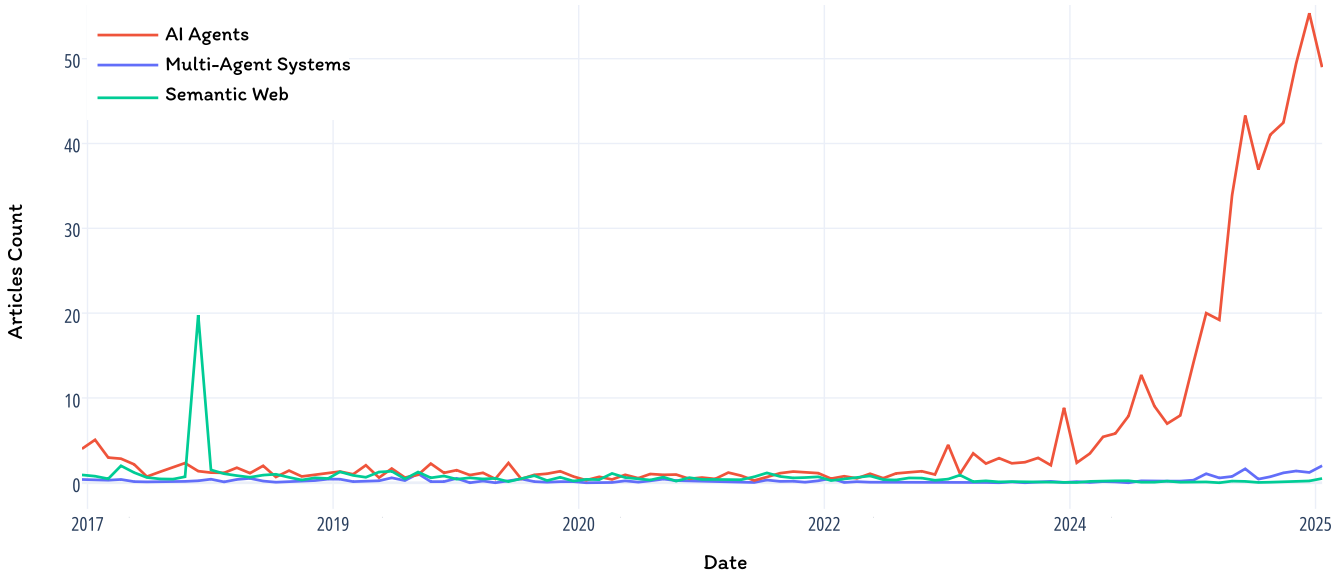

Figure 5. **GDELT (Global Database of Events, Language and Tone) topics 2017-2025 years evolution.** Public data from GDELT [101] presents world-wide monthly average articles count starting from 2017 regarding the specific phrases. Around January 2023 one can see a significant rise of AI Agents (red line). Meanwhile Semantic Web and Multi-Agent Systems topics are experiencing just a drop of social interest and still stay in the shadow.

[95]. The comparison of key characteristics between Borg and Kubernetes, along with their implications for agentic systems, is provided in Table 3.

4.2. LLMs and the Need for New Interaction Standards

The rise of powerful LLMs, with a rapid pace of development exemplified by models such as those from OpenAI (GPT-4.5 [6]), Google (Gemini 2.5 Pro [7]), Anthropic (Claude 3.7 Sonnet [8]), xAI (Grok-3 [9]), Mistral Large 2 [10]), and Alibaba (Qwen 3 [11]), has dramatically advanced the ability to understand and generate natural language, perform reasoning and planning, and even generate code [81], [88], [90]. As a result of these advances, AI agents can now tackle much more complex and unstructured tasks than before [90]. Simultaneously, new training paradigms have emerged to further improve agent reasoning and adaptability. Reinforcement learning techniques enable agents to optimize decisions over long action sequences [99], and imitation learning from human demonstrations provides strong priors for complex behaviors [100]. In addition, retrieval-augmented generation (RAG) methods allow agents to dynamically query external knowledge bases or documents in real time, ensuring their responses remain up-to-date and grounded in relevant information [91].

However, this new wave of AI agents face a major challenge: there is still no standardized methods for them to interact with external resources (such as websites, databases, APIs, and tools) or with one another. Consequently, developers have had to implement custom integrations for each agent and each external resource. In practice, these bespoke integrations rely on techniques like web scraping (e.g., parsing HTML via DOM or XPath), invoking RESTful APIs, or even controlling a headless browser with automation frameworks like Selenium to enable agents to access web content and services. This ad-hoc approach is neither efficient nor scalable and hinders the development of integrated multi-agent systems. This gap has created an urgent need for new protocols that can simplify and standardize such interactions, enabling LLM-based agents to realize their full potential in a web environment.

References to tool usage make it clear that an agent's power is measured by what it can do externally. Multi-agent systems show that collective intelligence emerges from interaction. MCP (see Section 4.3) standardizes the model-to-tool connection, while A2A (see Section 4.4) standardizes the agent-to-agent link. Consequently, an agent's identity and capabilities are increasingly tied to these external interfaces and its position within the "Web of Agents." Table 6 provides a comparison of the "AI-Agent" concept across the different paradigms discussed above. There is a clear shift from agents defined primarily by their internal structure and logic to agents whose nature and capabilities are increasingly determined by their ability to interact (with a broad ecosystem of tools, data, and other agents) via standardized protocols.

4.3. Model Context Protocol (MCP): Standardizing Model-to-Context Interaction

In response to this need, Anthropic unveiled and open-sourced the Model Context Protocol (MCP) in November 2024 [3]. MCP is positioned as an open standard, a universal interface for plugging AI models into external data sources and tools. The aim of MCP is to standardize how applications (acting as MCP hosts or clients) supply context

Table 3. COMPARATIVE ANALYSIS OF BORG AND KUBERNETES FOR AGENTIC SYSTEMS

Characteristic	Borg	Kubernetes	Implication for Agentic Systems
Primary Unit of Work	Job / Task (A process)	**Pod** (A group of co-located containers)	The **Pod** model is a natural fit for multi-component agents (e.g., core logic + logging sidecar), allowing them to be deployed and managed as a single atomic unit.
Networking Model	IP-per-machine. The orchestrator schedules ports as a resource.	IP-per-pod. Every pod gets its own IP address, creating a flat network.	Simplifies agent development and interoperability. Agents and their tools can be developed independently without needing to coordinate port usage, enabling a loosely-coupled ecosystem of "off-the-shelf" components.
Service Discovery	Ad-hoc systems built around Borg, often tied to specific naming services like Chubby.	First-class **Service** object. A stable endpoint for a dynamic set of pods selected via labels.	Provides a robust, built-in mechanism for agents to discover and communicate with each other and with necessary tools (e.g., databases, APIs) in a resilient way, as the **Service** abstracts away pod failures and rescheduling.
API Paradigm	Imperative, complex, and heterogeneous collection of APIs and configuration languages.	Declarative, consistent, and extensible REST API. Users define a "desired state".	Aligns conceptually with agent planning. An agent can declare the desired state of its required resources, and the platform is responsible for fulfillment. The extensible API allows the platform to be adapted with custom resources for agents.
Grouping & Selection	Rigid, implicit grouping by **Job**. Users embedded metadata in names.	Flexible, explicit grouping via arbitrary key/value **Labels** and **Selectors**.	Allows for dynamic and fine-grained management of agent populations. Operators can manage subsets of agents (e.g., "canary-release-agents," "data-analysis-agents") for targeted updates, monitoring, or A/B testing.
Primary Design Goal	Machine-Oriented: Maximize cluster resource utilization and efficiency.	Application-Oriented: Simplify the developer experience of building and managing distributed systems.	The application-oriented focus abstracts away infrastructure complexity, allowing agent developers to focus on agent logic rather than low-level operational details, accelerating development and fostering a richer ecosystem.

(data) and tools (actionable capabilities) to AI models, particularly LLMs [3]. This allows models to fetch up-to-date information needed for a given task and to trigger actions in external systems through a single, consistent interface.

The MCP architecture follows a Client–Host–Server pattern and uses the JSON-RPC 2.0 protocol for messaging [3]. In practice, Microsoft's NLWeb project exemplifies the MCP approach integrated directly into user-facing web applications [103]. Each NLWeb instance effectively acts as an MCP server embedded within a website, exposing that site's data and functionality through a standardized natural-language interface. This allows external AI agents (as well as human users) to discover and interact with the website's content and actions via MCP, demonstrating a real-world implementation of the Model Context Protocol in everyday web environments.

In the MCP architecture, the key roles are defined as follows [3]:

- **Host:** The AI-driven application (e.g., a chatbot or an IDE with an integrated assistant) that initiates

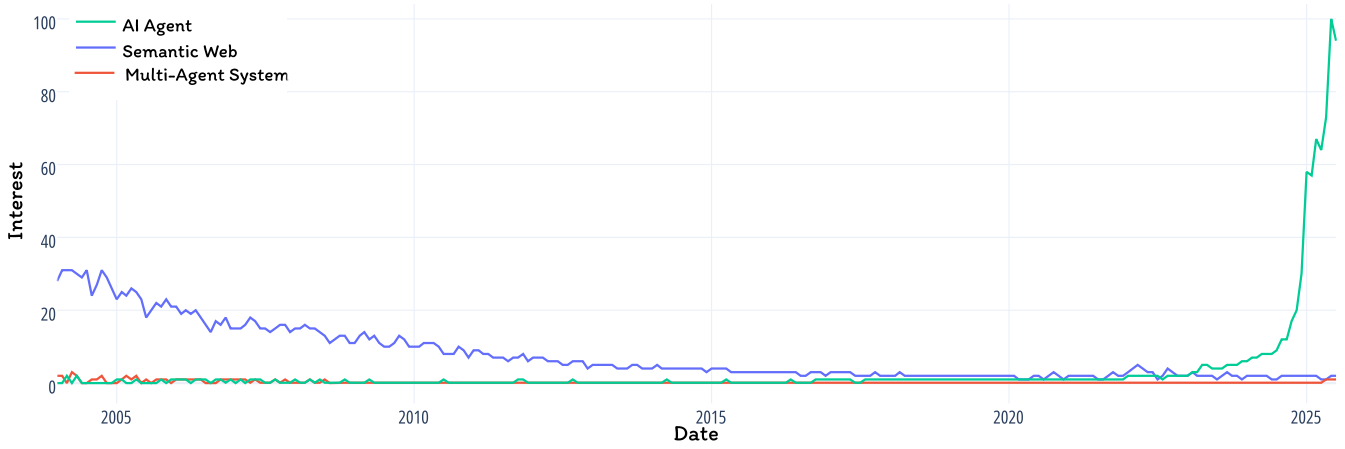

Figure 6. **Keywords interest based on Google Trends [102] data since 2004** Google Trends reflects the relative frequency of searches over the last 20 years and across the world, in particular the declining slope of Semantic Web (amethyst), the burst of interest for AI Agents (teal), and stable absence of interest for Multi-Agent Systems (red).

connections to an MCP server.
- **Client:** A component inside the host that maintains a connection to a single MCP server.
- **Server:** A lightweight service exposing particular data or capabilities via the MCP protocol.

MCP servers expose their capabilities to clients through a set of standardized primitives [3]:

- **Resources:** Structured data that the server can supply to enrich the model's context (e.g., document snippets, code, search results).
- **Prompts:** Predefined instructions or query templates that can guide the model's behavior or be presented to the user.
- **Tools:** Executable functions or actions that the model can invoke via the server (e.g., database queries, web searches, sending messages).

Key aspects of MCP [3], [104] include:

- *Standardization and Interoperability:* MCP replaces a tangle of bespoke integrations with a single protocol, collapsing the $M \times N$ integration problem (connecting M models with N tools) down to $M + N$.
- *Vendor Neutrality:* The protocol isn't tied to any particular model or AI provider.
- *Openness and Ecosystem:* MCP is an open standard with a public reference implementation, which has spurred a growing ecosystem of MCP servers for popular services like Google Drive, Slack, GitHub, databases, and more.
- *Security:* The specification emphasizes the need for explicit user consent before granting an agent access to data or invoking tools on a user's behalf.

Notably, MCP does not mandate the use of formal ontologies (such as RDF or OWL) to describe the semantics of resources or tools. Instead, meaning is conveyed through the structure and metadata of the JSON-RPC messages, rather than through any externally defined ontology or vector embedding space. This design choice simplifies implementation and offers flexibility, since new tool capabilities can be added by simply providing natural-language descriptions, instead of developing formal semantic models. However, it also limits machine-level semantic interoperability and precision, as meaning remains encoded in unstructured language rather than in a formal representation.

4.4. Agent-to-Agent (A2A) Protocol: Enabling Inter-Agent Communication

Where MCP addresses an agent's interaction with its environment (data and tools), the Agent-to-Agent (A2A) protocol (introduced by Google in April 2025 [89]) targets another key challenge: enabling communication and coordination among the agents themselves. A2A is also an open protocol, developed in collaboration with over 50 partners [89].

The goal of A2A is to allow AI agents, built on different frameworks and by different vendors, to communicate securely, exchange information, and coordinate their actions to tackle complex tasks that require joint effort. A2A is designed as a complement to MCP, not a replacement. If MCP is the "wrench" for tool access, then A2A is the "mechanics' dialogue" [89].

The A2A architecture also follows a client–server model, but applied to inter-agent interaction [89]:

- **Client Agent.** The agent that initiates a task and interacts with another agent (analogous to the client in a client–server exchange).
- **Remote Agent.** The agent that receives the task from the client agent and executes it (acting as the A2A server for that request).
- **Interaction Channel.** Communication built on widely adopted web standards, using HTTP(S) as transport, JSON-RPC 2.0 for structuring requests and responses, and Server-Sent Events (SSE) for streaming task-status updates [89].
- **Long-Running Tasks.** A2A supports tasks that may span minutes or hours by using a formal *Task* abstraction with built-in progress tracking. For example, a client agent can subscribe to updates via a `tasks/sendSubscribe` call, allowing the remote agent to push incremental status updates and partial results over SSE in real time. Alternatively, clients may poll using `tasks/get` or register webhook callbacks with `tasks/pushNotification/set` to receive asynchronous notifications when the task state changes or completes [89].

Key Concepts of A2A [89]:

- **Agent Card.** A standardized metadata file (typically in JSON and served at `/.well-known/agent.json`) that describes an agent's capabilities, its endpoint URL, and any authentication requirements. Agent Cards allow agents to advertise their skills and for others to discover available agents and their features.
- **Task.** The central unit of work in A2A. A client agent initiates a task, which carries a unique identifier and progresses through various states (e.g., `submitted`, `in_progress`, `awaiting_input`, `completed`, `failed`, `cancelled`).
- **Message.** The fundamental unit of communication between agents, structured similarly to messages in human–AI dialogues (with roles like "user" for the requesting agent and "assistant" or "agent" for the responding remote agent).

Key aspects of A2A [89] emphasize:

- **Built on Web Standards.** A2A is designed to simplify integration with existing IT and web infrastructure by using familiar protocols (HTTP, JSON-RPC,

SSE) rather than inventing new transport mechanisms.

- **Capability Discovery.** The standardized Agent Card mechanism allows agents to dynamically discover each other's capabilities and interfaces in a uniform way.
- **Security and Trust.** A2A includes built-in support for secure communication (TLS for transport, authentication tokens, etc.) and emphasizes access control, given that agents might be instructed to perform actions on behalf of users or other systems.

A2A does not appear to mandate the use of formal OWL/ontology definitions for agent capabilities. Similar to MCP's philosophy, it opts for a pragmatic, JSON-based description (Agent Cards) to facilitate discovery. Compared to classical agent communication languages like FIPA ACL or KQML, A2A shares the same inter-agent communication goals but adopts a more lightweight, web-native approach with declarative discovery and interoperability built in [63]. In effect, A2A aims to do for heterogeneous AI agents what protocols like HTTP did for heterogeneous web servers and clients.

4.5. Bridging MCP and A2A

Building upon the capabilities of the Model Context Protocol (MCP) (Section III.A) and the Agent-to-Agent (A2A) protocol (Section III.B), this subsection examines how these protocols jointly enhance the Web of Agents architecture. Each protocol offers complementary functionalities. MCP focuses on establishing a shared context among agents, whereas A2A provides a direct communication channel for inter-agent exchanges. Used in concert, these protocols complement each other to improve overall agent collaboration. In particular, the contextual information maintained through MCP can inform A2A message exchanges, ensuring consistent interpretation of messages by all agents. Conversely, insights obtained via A2A interactions can be fed back into the shared context through MCP, creating a feedback loop that improves the system's adaptability. As a result, the combined use of MCP and A2A yields a more coherent and flexible multi-agent system, a benefit that will be examined further in the comparative analysis to follow.

Prior studies have typically treated tool-oriented protocols and inter-agent communication in isolation. In contrast, we analyze how MCP and A2A can operate in synergy within an integrated agent ecosystem – an aspect not explored in earlier literature. For example, we highlight frameworks such as AutoGen and LangChain, which combine secure tool use via MCP with collaborative multi-agent workflows enabled by A2A. By examining these implementations and summarizing their capabilities in comparative tables, we provide novel insights into the complementary roles of these protocols in advancing multi-agent systems.

Design Patterns and Frameworks.. Contemporary agent architectures increasingly reflect this MCP–A2A synergy. A common pattern is an orchestrator agent coordinating a team of specialist agents [106]. The orchestrator delegates subtasks via inter-agent messaging (A2A) while each agent retrieves information or executes actions using standardized tool APIs (MCP). This decoupled design lets complex problems be decomposed: one agent focuses on planning and delegation, another handles database queries or computations on demand. Crucially, the A2A layer carries dialogue and task negotiation between agents, and the MCP layer injects the necessary external context or operations for each agent's task. The open-source AutoGen framework illustrates this interplay: it orchestrates multiple LLM-based agents that converse to solve problems, with agents dynamically invoking tools (e.g., web search, code execution) through a unified context interface [107], [114]. Likewise, LangChain provides abstractions for LLM agents to use a suite of tools via a standardized interface [93], and can be extended to multi-agent workflows where one agent's output feeds into another's input. The CrewAI toolkit explicitly models a "crew" of cooperating AI specialists, each leveraging plugins or APIs as needed [106].

Implications for Scalability and Ecosystem.. Combining horizontal and vertical integration in this manner greatly enhances scalability and modularity of agent systems. New specialized agents or tools can be introduced as plug-and-play components, so long as they adhere to the A2A or MCP standards. This loose coupling means an agent collective can be scaled out simply by adding more agents with new capabilities, without redesigning the core system. For instance, AWS Strands agent SDK explicitly supports both MCP and A2A, allowing developers to weave in new tool-using agents and inter-agent links in a cloud-native fashion [108]. Similarly, Microsoft's Semantic Kernel has been extended to incorporate A2A communications alongside MCP-compatible "skills", showing how platforms are aligning around both protocols [109], [110]. The broader vision is that standard interfaces will enable an agent marketplace: third-party services (exposed via A2A) can be discovered and invoked by any agent, while MCP ensures secure access to data/tools. Such an ecosystem, built on open protocols, promises unprecedented flexibility (agents can form ad-hoc coalitions, outsource subtasks to experts, and tackle problems beyond the scope of any single model). In short, MCP and A2A in tandem lay the groundwork for interoperable, scalable agent networks, where emergent behaviors arise from simple, well-defined interactions.

Limitations and Next Steps. The key advantage of MCP lies in its *structured API interface*. It is simpler and more robust for machines to access data or functionality via clearly defined tool calls rather than scraping human-facing interfaces. However, MCP is inherently *point-to-point*. It assumes that the calling agent already knows which tool or service to invoke. There is no built-in mechanism for service discovery or dynamic composition of multi-agent workflows. Additionally, MCP does not address higher-level concerns such as trust, pricing, or marketplace interactions.

However, A2A protocol provides the communication layer, but lacks several higher-level features. There is no built-in mechanism for discovering available agents or services on the open internet (A2A assumes you know the other

Table 4. COMPARISON OF FIPA ACL, MODEL CONTEXT PROTOCOL (MCP), AND AGENT-TO-AGENT PROTOCOL (A2A)

Characteristic	FIPA ACL	MCP	A2A
Primary purpose	Standardize communication among heterogeneous intelligent agents	Standardize AI-app interaction with external data sources and tools	Standardize communication, coordination, and discovery among independent AI agents
Key concepts	Performatives (speech acts), ontologies, interaction protocols, platform	Host, client, server, primitives (resources, hints, tools, sampling)	Client agent, remote agent, Agent Card, task, message, artifact
Communication style	Asynchronous performative messaging; requires a FIPA platform	Request–response and asynchronous interactions via JSON-RPC 2.0	Request–response (JSON-RPC), streaming (SSE), push notifications over HTTP
Semantics approach	Formal modal-logic semantics; explicit ontologies required	Informal: semantics defined by JSON structure; no formal ontologies required	Informal: capability semantics defined in Agent Card JSON; no formal ontologies required
Underlying Theory	Speech-act theory, BDI logic [29]	Web architecture (REST/RPC principles)	Web architecture (REST/RPC principles)
Semantic Expressiveness	High (formal semantics based on speech-act theory)	Low (RPC-style, informal semantics)	Low (RPC-style, informal semantics)
Capability discovery	Via Directory Facilitator (DF) service of the FIPA platform	Server advertises supported capabilities on connect	Via standardized Agent Card (‘/.well-known/agent.json‘)
Primary Overhead	Implementation complexity, platform dependency	Network latency, lack of formal guarantees	Network latency, lack of formal guarantees
Key features	Rich communication semantics, platform standardization, formal foundation	Vendor neutrality, openness, ease of integration (M + N), growing ecosystem	Built on web standards, model-agnostic, built-in security, complements MCP
Limitations / challenges	Complexity, practical interoperability issues, disconnect from web standards	Early-stage, community adoption dependent, tool-security concerns	Early-stage, community adoption dependent, task-management complexity
Relation to Web / SW	Requires middleware; conceptual linkage to Semantic Web via ontologies	Built on web protocols (JSON-RPC); not tied to formal SW semantics	Built on HTTP, JSON-RPC, SSE; not tied to formal SW semantics

agent's address or identity ahead of time). It also does not inherently deal with economic incentives or marketplaces, for example, how to bid or pay for agent services is outside the scope of A2A. Furthermore, A2A by itself does not establish trust, reputation, or legal accountability frameworks between agents. It focuses on enabling the technical conversation.

A2A is a crucial piece (allowing heterogeneous agents to interoperate), but additional layers are needed to achieve a web-like ecosystem where an agent can autonomously find the service it needs and engage with it under agreed-upon terms.

4.6. Comparative Analysis of Interaction Protocols

To better understand the evolution and differences in approaches to enabling agent interoperability, it is useful to compare FIPA ACL, MCP, and A2A across key characteristics (see Table 4).

Table 4 clearly illustrates how these approaches have evolved. FIPA ACL was an ambitious attempt to create a rich, formally grounded language for agents, but it encountered real-world integration challenges [23], [69]. MCP and A2A, emerging in the LLM era, adopted a more pragmatic approach, focusing on specific interaction patterns (model-to-context and agent-to-agent, respectively) and leveraging standard Web technologies [3], [89]. This pragmatism likely lowers the barrier to entry for developers and accelerates adoption. This evolution highlights a central architectural trade-off. FIPA ACL, with its foundation in speech-act theory and formal BDI logic, offered high semantic expressiveness, enabling agents to engage in complex, stateful negotiations with unambiguous intent [30], [63]. However, this richness came at the cost of significant implementation complexity and a tight coupling to heavyweight platforms, which ultimately hindered widespread adoption. In contrast, MCP and A2A prioritize simplicity and developer experience by using a lightweight, RPC-style interaction model. While this approach is less semantically expressive and lacks the formal guarantees of FIPA, its reliance on ubiquitous web standards dramatically lowers the barrier to entry, fostering a more accessible and scalable ecosystem.

MCP and A2A serve different yet complementary roles in enabling a functional Web of Agents. MCP standardizes how an AI model connects to data and tools, whereas A2A governs communication and coordination between agents [87], [88]. This marks a departure from the original Semantic Web vision, where semantics were intended to be embedded directly within the data [1].

4.7. Representative LLM-Agent Frameworks

The rapid growth and adoption of these agent frameworks can be quantitatively observed through their community engagement metrics. Figure 7 illustrates the evolution of GitHub stars for major open-source agent frameworks over a two-year period, demonstrating the exponential growth in developer interest and adoption. This data provides empirical evidence of the increasing importance of multi-agent systems in the AI ecosystem.

As shown in Figure 7, the ecosystem has experienced remarkable growth since early 2023, with LangChain emerging as the clear leader, accumulating over 95,000 stars by mid-2025. Within the modern Web of Agents paradigm, the emergence of AutoGPT and its successors has marked an important step in operationalizing autonomous LLM-driven agents. These systems demonstrate how the abstract concept of an agent that can interpret context, plan, and act is realized in practice, serving as practical implementations of LLM-based agents. AutoGPT (2023) [38] was one of the first publicly visible examples of an autonomous agent powered by an LLM, capable of setting its own sub-goals, invoking tools (e.g. web search, file I/O), and iteratively refining its approach to achieve a high-level objective. The viral success of AutoGPT underscored the potential of

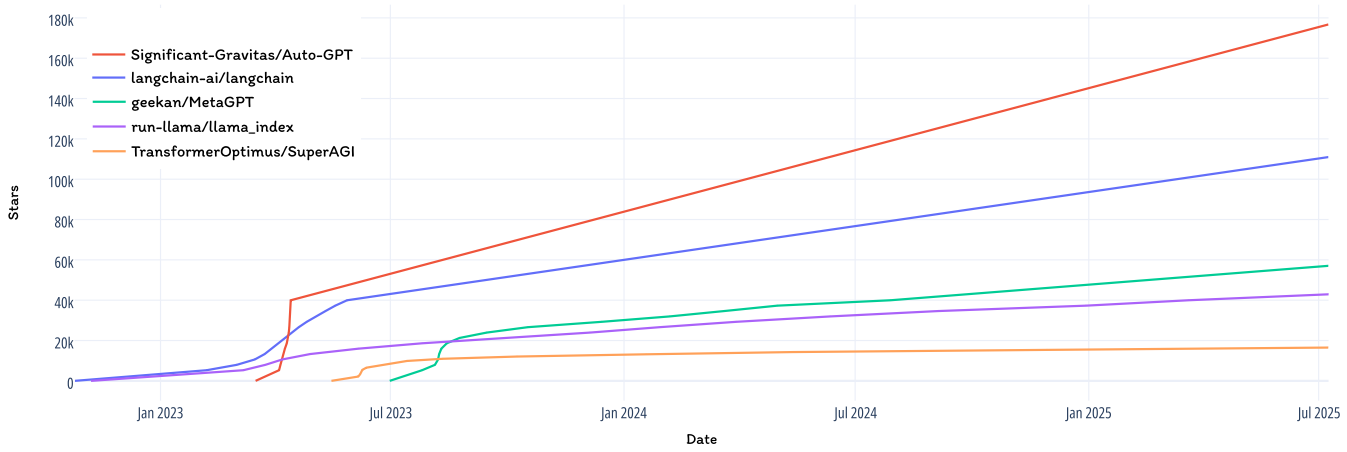

Figure 7. **GitHub stars growth dynamics for major AI agent frameworks.** Cumulative star count evolution for five prominent open-source agent repositories (LangChain, Auto-GPT, MetaGPT, LlamaIndex, and SuperAGI) from January 2023 to July 2025. The stacked area chart demonstrates the relative popularity and adoption rates, with LangChain showing the most significant growth trajectory, followed by Auto-GPT. The visualization highlights the accelerating community interest in agent-based AI systems, particularly after the release of GPT-4 in March 2023.

such agents and catalyzed the development of a myriad of frameworks that build upon similar ideas to make LLM-agent construction more accessible. These frameworks are crucial in the evolution of the Web of Agents because they encapsulate the complexities of LLM usage (prompt orchestration, memory management, tool integration) into reusable modules, enabling developers to more easily create sophisticated agents. These frameworks effectively connect the theoretical architectures of agent systems with real-world applications by offering ready-to-use components built around large language model capabilities.

Several prominent LLM-agent frameworks have followed AutoGPT's lead. The development of these frameworks reveals a clear evolutionary trajectory, moving from monolithic agent designs toward a more modular and collaborative architectural stack. This progression can be conceptualized as a stack with a data layer, an application logic layer, and a collaboration layer, reflecting a rapid maturation of the field.

- **LangChain** [94] introduced a comprehensive toolkit for developing LLM-powered applications, offering abstractions for chaining prompts, managing conversational memory, and integrating external tools into an agent's reasoning loop.
- **LlamaIndex** [111] is an open-source data framework specifically engineered to connect LLMs with custom, private, or domain-specific data sources, with a core focus on enabling robust Retrieval-Augmented Generation (RAG). LlamaIndex is not a general-purpose agent framework but rather an indispensable data layer or specialized tool that agents built with frameworks like LangChain can leverage. It pragmatically solves the critical challenge of grounding agent reasoning in reliable, external knowledge, a goal central to the original Semantic Web vision. Yet, it embodies the modern "semantics in the model" paradigm; instead of relying on formal ontologies like RDF/OWL, it uses the LLM's implicit understanding of text, augmented by vector retrieval, to interpret data. Its sophisticated capabilities for parsing complex, nested documents have made it a premier solution for enterprise RAG pipelines, particularly in sectors like finance and law. .
- **SuperAGI** [112], released as an open-source "dev-first" platform, focuses on the orchestration and management of autonomous agents, with an emphasis on extensibility (via plugins) and robust execution of long-running tasks.
- **MetaGPT** [113] explores a multi-agent collaboration paradigm: it allocates multiple specialized LLM agents to different roles in a problem (e.g. planner, coder, tester), and coordinates their interactions through predefined Standard Operating Procedures (SOPs) to tackle complex tasks (notably in software engineering scenarios).

The ecosystem of agent frameworks is rapidly expanding and includes many other notable open-source projects, such as SuperAGI [112], which focuses on building and managing autonomous agents for long-running tasks. Or AutoGen [114], which enables the development of LLM applications using multiple agents that can converse with each other to solve tasks through automated conversations and human feedback. And CrewAI [115], which orchestrates role-playing agents that collaborate to achieve a common goal. These frameworks can be broadly categorized by their licensing and distribution model, as shown in Table 5.

All of these tools share the common goal of simplifying the creation of LLM-based agents and workflows, effectively acting as building blocks for a practical Web of Agents. Table 5 summarizes key functional characteristics of AutoGPT and some of its prominent successors. These frameworks also complement emerging protocols (MCP,

Table 5. CLASSIFICATION OF REPRESENTATIVE LLM-AGENT FRAMEWORKS BY LICENSING MODEL.

Open Source Frameworks	Proprietary / Platform-based
LangChain [94]	OpenAI Assistants API [116]
MetaGPT [117]	Google Vertex AI Agent Builder [118]
AutoGen [114]	Amazon Bedrock Agents [119]
LlamaIndex [111]	Microsoft Azure AI Agent Service [120]
SuperAGI [112]	Anthropic Claude Computer Use [121]
CrewAI [115]	IBM Watson Orchestrate [122]

Table 6. COMPARISON OF REPRESENTATIVE LLM-BASED AGENT FRAMEWORKS

Characteristic	AutoGPT (2023) (Single LLM Agent)	LangChain (2022) (LLM App Framework)	SuperAGI (2023) (Agent Platform)	MetaGPT (2023) (Multi-Agent System)
Primary focus	Fully autonomous goal-driven agent; demonstrate LLM autonomy in executing tasks end-to-end.	Developer library for building LLM-powered applications (chains, agents, chatbots, etc.) with modular components.	Production-ready framework to deploy and manage useful AI agents (concurrent or long-running) for real-world tasks.	Research framework for collaborative multi-agent workflows, simulating a team of specialized LLM agents to tackle complex projects.
Level of autonomy	High. Runs continuously without user intervention, deciding and executing next actions until goal completion; single-agent recursive loop (self-feedback).	Configurable. Not inherently autonomous; supports synchronous chains or custom agent loops defined by developer.	High – designed for autonomous operation, though user can oversee via an interface; supports concurrent independent agent instances.	High (multi-agent). Agents autonomously exchange information and outputs; once initialized, system self-drives through task hand-offs.
Tool/Plugin integration	Built-in plugins for web search, web browsing, file I/O, and code execution; extensible via custom plugin development.	Extensive support for tools/APIs. Pre-built connectors for search, databases, and custom APIs.	Plugin marketplace with ready-made toolkits for common tasks. Configurable plugin extensions.	Focuses on inter-agent communication and artifact generation. Limited general external API/plugin integration by default.
Scalability & deployment	Intended as a personal agent. Local process loops on API calls. Manual parallelism required for scaling; lacks built-in monitoring or failover.	Library-level scalability in user-managed infrastructure. No inherent multi-agent coordination; developer-managed deployment.	Designed for scalable deployment. Supports concurrent agent instances, includes web UI and logging for monitoring. Suitable for enterprise (Docker/cloud).	Research prototype. Resource-intensive multi-agent setup. Not optimized for high-throughput tasks; requires significant computational resources.
Task orchestration	Single-agent loop with GPT-driven planning and execution via ReAct-like feedback.	Developer-managed orchestration via chains or custom control flows in code.	Orchestrator layer for managing agent workflows with event-driven architecture and pre-defined ReAct sequences.	Role-based pipeline using a publish/subscribe message bus. SOP-driven handoff between specialized agents.
MCP/A2A integration	Experimental community efforts for A2A; no official support yet. Potential for MCP tool invocation via JSON-RPC.	Supports emerging standards. Listed as A2A partner. Potential MCP client for tool calls through generic tool integration interface.	Compatible with JSON-RPC tool APIs (MCP). Extendable to A2A though not yet documented.	Implements a custom message-passing protocol analogous to A2A. Potential future integration with MCP/A2A for greater interoperability.

A2A) by focusing on the agent's internal decision-making and capabilities, potentially interfacing with such standards for external communication in the future.

Table 6 summarizes key characteristics of AutoGPT and these successor frameworks. All of these tools share the common goal of simplifying the creation of LLM-based agents and workflows, effectively acting as building blocks for a practical Web of Agents. They also complement emerging protocols (MCP, A2A) by focusing on the agent's internal decision-making and capabilities, potentially interfacing with such standards for external communication in the future.

5. Evolution of the Web of Agents Vision: From the Semantic Web to Modern Agent Ecosystems

The concept of the Web of Agents has come a long way, evolving from its origins in ideas deeply entwined with the Semantic Web to the modern ecosystems shaped by LLMs and new interaction protocols. Examining this progression highlights both the enduring core goal and the significant shifts in how that goal is pursued.

5.1. Development Trajectory: Continuities and Divergences

The fundamental goal has remained constant: to build a Web in which autonomous software agents can seamlessly interact with resources and with one another to perform tasks on behalf of users [1], [20], [35]. This aim can be traced from the earliest Semantic Web scenarios all the way through to today's applications powered by LLM-based agents [23], [87].

However, the paths to achieving this goal have shifted dramatically:

- **Early vision (Semantic Web).** Relied on embedding explicit, formal semantics into Web data using RDF and OWL standards. Agent intelligence was expected to stem from logical inference over these structured datasets. Interoperability was to be

achieved through shared ontologies and, where appropriate, standardized agent communication languages like FIPA ACL. We define this approach as "semantics in the data", where meaning is explicitly encoded in formal structures. [1], [19], [22], [77]

- **Modern approach (LLMs + Protocols).** Leverages powerful large language models for implicit understanding of both structured and unstructured data and for executing complex reasoning and planning. Interoperability is provided by standardized interaction protocols (MCP, A2A) built on established Web standards (HTTP, JSON-RPC, SSE). Formal RDF/OWL-style semantics aren't baked into the protocols themselves; most of the semantic heavy lifting happens inside the LLM. We might label this "semantics in the model." [3], [81], [87]–[90], [92]

This shift reflects a broader paradigm change in AI, from knowledge- and logic-based systems to data-driven, deep-learning architectures. LLMs have sidestepped the challenge of creating an exhaustive, formal semantic layer on the Web by providing models that can extract meaning directly from the existing, less-structured content.

5.2. Comparison of Early Visions and Modern Approaches

By comparing use cases, we can spot both overlaps and distinctions. Tasks such as scheduling meetings or automating purchases, envisioned in early work, fit perfectly with what we expect modern agents to handle [19], [35], [86], [123], [124]. However, the mechanisms they use to accomplish these tasks today are fundamentally different.

A Semantic Web agent charged with scheduling a meeting would first locate each participant's semantically annotated calendars [39], then rely on a time-and-event ontology to interpret availability [125]. It might interact with a dedicated booking service via FIPA ACL [62], using logical inference over those structured data sources to resolve any conflicts.

A modern LLM-powered agent tackling the same task would instead lean on its natural-language understanding to parse the request [87], call calendar APIs (perhaps through an MCP-compliant tool [3]) compose and send queries, and interpret the responses using its internal reasoning models for planning and coordination [88]. If it needed to negotiate details with another user's agent, it could fall back on the A2A protocol to exchange messages and finalize the schedule [2].

The core difference lies in where the "intelligence" resides and how mutual understanding is achieved. In the early Semantic Web approach, intelligence was distributed across formal data structures and inference engines, with interoperability rooted in shared ontologies [22], [62]. In the modern workflow, intelligence is centralized in the LLM itself, while interoperability depends on standardized protocols [3], [89], and the depth of semantic alignment hinges on the LLM's capabilities and the quality of protocol descriptions [3].

5.3. Achieved Progress

There is no doubt that modern approaches demonstrate significant progress:

- **Enhanced AI capabilities.** LLMs provide far more flexible and powerful natural-language understanding, "common-sense" reasoning, planning, and adaptability to diverse tasks and unstructured information than was previously possible [87], [88], [90].
- **Pragmatic protocols.** MCP and A2A offer standardized solutions to key integration challenges (accessing context/tools and inter-agent communication) using widely adopted Web technologies, which simplifies their adoption [3], [89].
- **Growing ecosystems.** The rapid emergence and popularity of frameworks around these protocols, such as LangChain for chain-of-thought orchestration and SuperAGI for autonomous workflows, attests to strong community demand [94], [126].
- **User-facing interfaces.** The WoA paradigm is now manifesting in end-user applications. Microsoft's NLWeb enables websites to offer natural-language query interfaces via an MCP endpoint [103], so that their content can be directly accessed and utilized by both human users and AI agents. Likewise, Magentic-UI is a visual human-centered web agent interface that collaborates with users on complex online tasks [127], transparently displaying its step-by-step actions in a dedicated panel while prioritizing user consent for any irreversible operation. These developments illustrate how WoA concepts are being implemented at the user-interface level, effectively bridging advanced agent capabilities with intuitive user experiences.

This progress has made the vision of the Web of Agents more tangible and practical than ever before. The current approach may well be a "good enough" solution that, thanks to the power of LLMs and the pragmatism of the protocols, can overcome the barriers that stalled earlier, more ambitious, but also more complex initiatives.

6. Persistent Challenges

The persistent challenges of creating a truly open Web of Agents are underscored by the current industry trajectory towards centralized, proprietary ecosystems. Major technology providers are launching "walled garden" agent marketplaces, such as Amazon's marketplace on Bedrock [128], OpenAI's GPT Store [129], and Google's Vertex AI Agent Builder [130]. These platforms, while accelerating agent deployment, sidestep the hard problems of decentralized trust, identity, and economic interoperability by imposing platform-centric governance.

This trend highlights the critical need for research into decentralized alternatives. These include architectural concepts aiming for open interoperability [131], economic models exploring agent-centric auction mechanisms [132], and

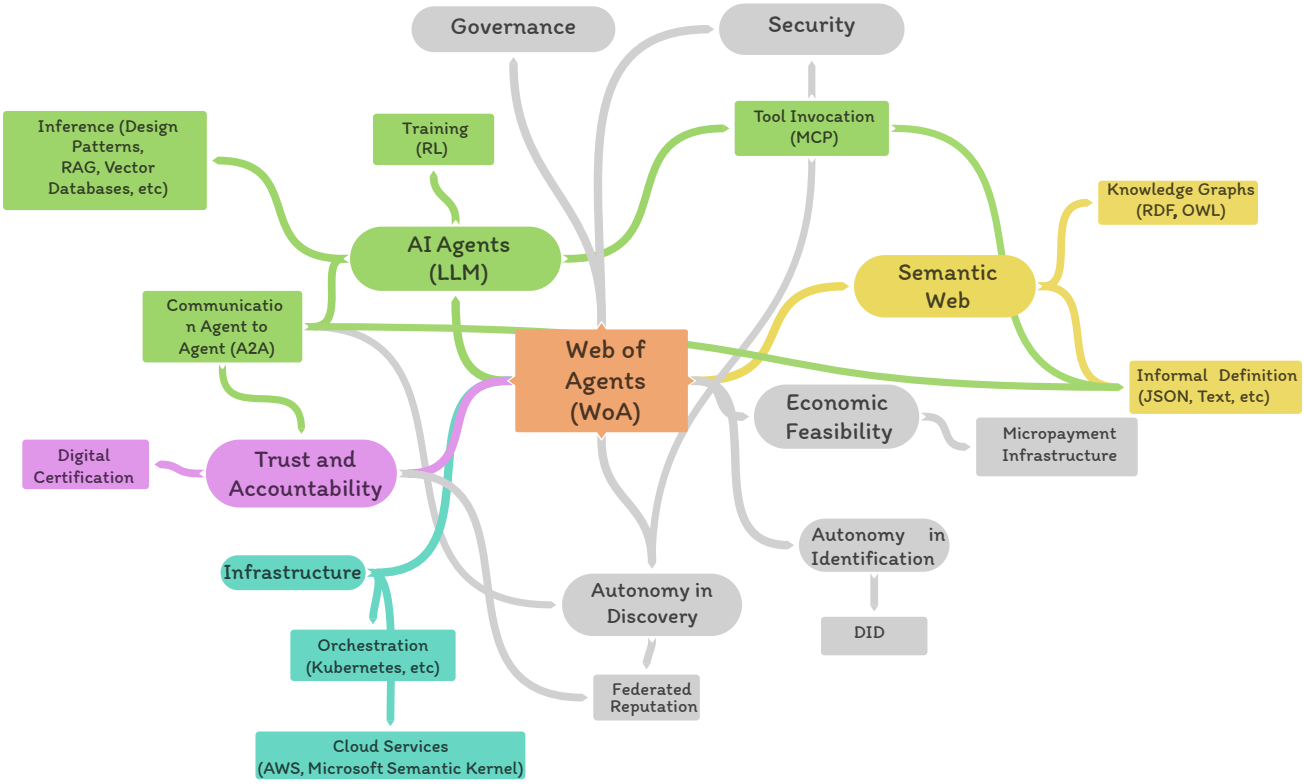

Figure 8. **A Conceptual Map of the Web of Agents (WoA) Ecosystem, Highlighting Established Components and Unsolved Challenges.** This map illustrates the key technologies and concepts that constitute the Web of Agents. The colored elements represent components that are relatively well-established, while the grey elements signify persistent socio-technical challenges that define the research frontier.**Core Components (Colored):** At the center are **AI Agents**, powered by Large Language Models (LLMs), which are enhanced through training (e.g., RL) and advanced inference techniques like Retrieval-Augmented Generation (RAG). These agents operate on a foundation of **Infrastructure**, including cloud services (like AWS and Microsoft Semantic Kernel) and orchestration platforms (like Kubernetes). Interaction is enabled by two key protocol types: **Tool Invocation** (e.g., MCP) for accessing external capabilities and **Communication** (e.g., A2A) for agent-to-agent dialogue. This communication builds toward **Trust and Accountability**, supported by mechanisms like digital certification. The **Semantic Web** provides the basis for shared meaning, which has evolved from formal **Knowledge Graphs** (RDF/OWL) in early systems to the **Informal Definitions** (JSON, natural language text) used in modern protocols.**Unsolved Challenges (Grey):** To achieve a truly open and robust WoA, the ecosystem must address several critical issues: **Autonomy in Identification and Discovery**: creating decentralized identity systems (e.g., DIDs) and federated reputation networks so agents can find and trust one another; **Economic Feasibility**: designing micropayment infrastructures and incentive models to ensure the ecosystem is self-sustaining; **Security and Governance**: developing robust defenses against new attack vectors and establishing clear frameworks for accountability, liability, and ethical alignment.

robust governance systems leveraging DAOs for dispute resolution [133]. Without such alternatives, the agent ecosystem risks balkanization and dependence on a few dominant, centralized entities.

While the need for a comprehensive alternative is clear, the current landscape of solutions is fragmented. At present, protocols such as MCP and A2A for connectivity, proposals like Agent.js to interface with existing sites, and lessons from the Semantic Web experience about formality and interface design each contribute pieces of the puzzle. However, none of these alone satisfies the requirements of an open ecosystem in which agents can discover, negotiate, and execute tasks across the Web autonomously. They either handle communication but not discovery or economic coordination (A2A, MCP), or provide a limited workaround for interaction without offering a scalable solution (Agent.js), or prove too rigid for practical deployment (classical Semantic Web ontologies). The question therefore remains: how can these insights be combined into a true Web of Agents? where agents can roam freely across Internet services? This section outlines the key components and main challenges that must be addressed.

Example. If a user asks an AI assistant to plan a trip to conference in London, the agent ought to be able to:

1) Register for conference;
2) Pay attendance fee applying all the discounts for IEEE members;
3) Find a right flight taking into the account frequent flyers membership price and preference (window or aisle)
4) Book hotel close to venue taking into

account's past travel history, membership and price;
5) Book a transfer from airport or prepare iternary.
6) Provide an iternary and itemized user plan, confirmation numbers, phones, etc.

Crucially, this chain of actions must occur "in the wild". That is, agents must operate on the open Internet, cooperating or competing with one another, rather than relying on a closed ecosystem or a fixed set of pre-defined APIs.

6.1. Trust, Accountability and Identity in Decentralized Networks

Autonomy in Discovery and Tool Use. An agent should be able to search for the tools or services it needs at runtime, rather than relying solely on pre-integrated endpoints. In other words, a Web of Agents must include an open-ended discovery layer: when an agent faces a novel task, it should be able to query a registry or search engine of agents and services, identify suitable candidates, and dynamically select collaborators to accomplish the job. This mirrors how new websites become indexed for human users, but it must be designed from the ground up for machine-to-machine interaction rather than human-readable content. Without such mechanisms, an agent cannot learn about newly published services in real time, which severely limits autonomy and scalability.

The fundamental identity crisis of autonomous agents requires sophisticated frameworks that extend beyond traditional identity management systems designed for human users or static machines. Self-Sovereign Identity (SSI) has emerged as the leading paradigm for addressing this challenge. Current implementations leverage Decentralized Identifiers (DIDs) following W3C specifications, with agent-specific methods including did:key for cryptographic identities, did:ion for blockchain anchoring, and did:dht for scalable distributed networks. Verifiable Credentials (VCs) enable cryptographically secured capability attestation, allowing agents to prove their authorization for specific actions without revealing unnecessary information [134].

While no single, all-encompassing protocol yet exists, current research addresses these challenges through complementary approaches. The development of comprehensive security architectures is a critical step, with proposals like Zero-Trust frameworks that ground agent security in decentralized identity and context-aware, fine-grained access control [135]. Concurrently, the principles of Intent-Based Networking (IBN) are being explored to create more autonomous and goal-oriented communication protocols [136]. To ensure long-term security, these systems must also anticipate future threats by integrating post-quantum cryptography, an area of active research for securing interconnected devices against quantum attacks [137]. Finally, the complex challenge of embedding ethical behavior is being addressed through foundational work in computational ethics, such as voting-based systems designed to align machine decisions with collective human norms [138], which serves as an early step toward creating verifiable and ethical agent conduct.

6.2. Economic Models and Incentive Alignment

Economic Feasibility and Incentives. An open agent economy must be grounded in economic reality. On the human Web, services carry explicit costs, and businesses have clear incentives. Likewise, agent services will need to charge for their usage, especially as complex multi-agent systems can consume significantly more computational resources than simpler models [139]. In practice, every agent-to-agent interaction may incur a micro-cost or reward. Embedding frictionless payment and pricing mechanisms into the fabric of agent interactions is essential to ensure that service providers are compensated and that resources are allocated efficiently.

Tokenization, the representation of service access or governance rights as digital tokens, provides the necessary economic primitives. However, designing a stable token-based economy requires formal modeling to prevent instability [140]. A major unresolved challenge is the high price volatility of the underlying crypto-assets, which creates significant uncertainty for service pricing and complicates long-term agent collaborations [141].

Reputation. In open agent ecosystems, reputation mechanisms are crucial for fostering trust and mitigating information asymmetry. Economically, a robust reputation system can prevent adverse selection and the classic "market for lemons" scenario, wherein quality uncertainty causes markets to collapse without trust signals [142]. By allowing past honest behavior and quality service to be observed, reputation serves as a proxy for reliability, aligning incentives for agents to maintain good standing. Indeed, trust and reputation models have long been studied as tools to reduce uncertainty among self-interested agents in decentralized environments [143], [144].

The "market for lemons" problem requires sophisticated reputation systems to prevent adverse selection. The FIRE (Flexible Integrated Reputation and Trust) model incorporates interaction trust, role-based trust, witness reputation, and certified reputation to provide comprehensive trust assessment in open multi-agent systems [145]. Context-aware trust models like FOCET (Functional Ontology of Context for Evaluating Trust) enable adjustable context weighting for improved total profit in dynamic environments.

Technically, implementing reputation without a central authority involves federated and cryptographic approaches. To manage trust in open, decentralized environments, various algorithmic frameworks have been proposed. A foundational example is the EigenTrust algorithm [146], which computes global reputation scores for peers in a P2P network by aggregating local trust values in a distributed and secure manner, effectively isolating malicious actors. Another approach uses self-sovereign identity and verifiable credentials: after each transaction, agents issue digitally signed feedback tokens (verifiable credentials) that attest to

performance; these credentials, tied to decentralized identifiers (DIDs) and signed by the issuer's key, enable any agent to verify trustworthiness without a central registry [147]. In peer-to-peer systems, algorithms like EigenTrust compute global trust scores by aggregating local feedback, reducing interactions with malicious nodes [146]. However, ensuring honest ratings remains challenging: incentive-compatible schemes reward truthful reporting and penalize dishonest feedback [148], while defenses against Sybil attacks and identity whitewashing include binding reputation to persistent identities or imposing costs on new identities (e.g., proof-of-work) [149]. Some proposals even leverage blockchain to store immutable feedback records, though on-chain solutions face scalability and off-chain verification issues [150]. A robust reputation infrastructure, combining these economic and technical strategies is essential for trust in a decentralized Web of Agents.

Adapting Business Models. The introduction of autonomous agents as service providers disrupts traditional web business models. One consequence is that digital marketplaces may evolve into agent-to-agent bidding platforms, where consumer representing agents and provider agents negotiate terms autonomously. Traditional advertising-based revenue models are also likely to be upended: because autonomous agents do not consume promotional content in the same way humans do, service providers may shift toward performance-based fees or usage-based API pricing, aligning revenue directly with delivered outcomes rather than impressions or clicks. Under this paradigm, the emphasis in service design shifts from consumer-facing interfaces to API-first architectures: businesses prioritize exposing functionality through standardized, machine-readable endpoints because their primary customers are software agents instead of human users. Competitive advantage will hinge on delivering superior value-efficiency, cost-efficiency, and reliability to agent consumers, rather than capturing human attention through banners or pop-ups.

API-oriented business models have evolved into two primary patterns: Partner API models enabling project-specific collaboration between agents, and Standardized API models providing wide customer base access through standardized interfaces. Monetization strategies include API-as-a-Product direct charging, subscription-based access with tiered pricing, and value network development through ecosystem network effects [151], [152].

6.3. Security and Resilience in Adversarial Environments

Trust, Security, and Accountability. For agents to act on behalf of users in an open-internet environment, security, privacy, and liability must be addressed comprehensively. Key questions include:

- How can an agent verify that a service agent is competent and not malicious?
- Who bears responsibility if an agent causes harm or fails to fulfill its contract?
- How can agents principal private information be protected when handled autonomously by agents?

A viable Web of Agents requires robust trust and reputation systems, certification mechanisms for agent services, and possibly legal or quasi-legal frameworks to enforce accountability. For instance, agents might carry digital certificates attesting to their identity, professional qualifications, or performance guarantees.

Indirect prompt injection attacks represent the most sophisticated threat vector against LLM agents, with the 2023 HouYi framework successfully exploiting 31 out of 36 real-world applications [153]. These attacks embed malicious instructions within external data sources, creating document infiltration through contaminated emails and web pages, multimodal exploitation using steganography in images, and RAG poisoning through malicious documents in knowledge bases.

The vulnerability of machine learning models to data poisoning attacks has been analyzed from numerous perspectives. A key methodological advance came from Koh and Liang [154], who adapted statistical influence functions to trace a model's predictions back to its training data. Their work demonstrates that by identifying the most influential training examples for a given prediction, it is possible to craft highly efficient and targeted training-set attacks, effectively demonstrating a surgical method for data poisoning rather than relying on brute-force data contamination.

Excessive agency exploitation manifests through three primary vectors identified in OWASP Top 10 (2025): excessive functionality where agents access unnecessary tools, excessive permissions with overly broad system access, and excessive autonomy lacking adequate oversight [155]. Attack scenarios include privilege escalation, lateral movement through connected systems, resource abuse via overwhelming API calls, and unauthorized data exfiltration.

Similarly, agents could be empowered to analyze and agree to Terms of Service (TOS) on behalf of their principals, automating a legally significant but often overlooked task for users. For such an agreement to be legally binding, it must be provable and attributable to an authorized party, even if that party is an AI agent acting as an instrument for its owner [156]. This process could drive a market-led standardization of service terms to make them more machine-readable. Furthermore, privacy can be preserved using cryptographic methods like Zero-Knowledge Proofs (ZKPs) [157]. A ZKP is a protocol where one party (the prover) can prove to another (the verifier) that they know a value or satisfy a statement, without conveying any information apart from the fact that the statement is true. For example, an agent could use a ZKP to prove it has the principal's authorization to agree to the terms, and even prove that the principal meets certain criteria required by the TOS (e.g., is from an eligible jurisdiction), all without revealing the principal's actual identity. This creates a verifiable and legally binding agreement while robustly protecting user privacy [158], [159].

Insurance schemes could underwrite agents against certain categories of failure or misconduct, so that compensated parties can recover losses if an agent acts negligently or maliciously. The existence and terms of such insurance could become a verifiable attribute of a service agent, incorporated into trust and risk assessment mechanisms [160]. This attribute could then be factored into service discovery and bidding processes, allowing a user's agent to prioritize services that offer better financial protection or to negotiate terms based on the level of risk coverage provided.

Strong authentication and authorization protocols are also essential. In a typical interaction, a user (the principal) delegates a task to their agent, often with a specific scope and budget, to achieve a goal [161]. The user agent may then need to interact with a remote service agent, necessitating two forms of authentication: agent-to-agent authentication to verify the identities of the interacting agents [161], and principal-to-service authentication to verify the authority of the original user who delegated the task [162]. Modern privacy regulations like GDPR create an opportunity to use advanced cryptographic methods. Zero-Knowledge Proofs (ZKPs), for instance, allow a user or agent to prove an attribute is true (e.g., being over 18 or having sufficient funds for a transaction) without revealing the sensitive underlying data [162]. This enables strong, privacy-preserving authentication of the principal when their agent interacts with a service, ensuring compliance and protecting user data [162].

6.4. Governance and Ethical Alignment

The ETHOS framework (Ethical Technology and Holistic Oversight System) establishes the first comprehensive governance model specifically designed for autonomous AI agents through blockchain-based global registries and smart contract automation [163]. ETHOS categorizes agents into four risk tiers (unacceptable, high, moderate, and minimal) with corresponding compliance requirements enforced through zero-knowledge proofs and Soulbound Tokens for verifiable credentials.

The framework introduces AI legal entities enabling autonomous systems to assume limited liability while ensuring accountability through insurance and compliance monitoring. Decentralized Autonomous Organizations (DAOs) provide multi-stakeholder governance structures involving developers, regulators, auditors, and ethicists in participatory decision-making processes. Decentralized justice and audit mechanisms require multi-stakeholder ecosystems incorporating internal company audits, independent third-party assessments, and participatory community oversight [164].

Economic Models and Payments. While an agent can be delegated access to a user's credit card for primary services like airline tickets or hotel bookings, a truly autonomous agent must also handle payments for a range of auxiliary services. These include fees for service discovery, charges from other specialized agents (e.g., for a legal analysis of Terms of Service), or even advertising costs.

For the agent ecosystem to be economically viable and self-sustaining, these operational costs must be accounted for. A robust economic model requires a mechanism to shift these costs to the end-user rather than burdening the service providers. The nature of these interactions, which often involve nanopayments rather than just micropayments, challenges the suitability of traditional, strictly transactional systems. This necessitates a move toward more flexible, eventually consistent payment models.

For instance, mechanisms like asynchronous rebalancing, as demonstrated in the Cycle protocol, allow participants to exchange payment information and rebalance channels without freezing them. This ensures continuity of service and minimizes failed payments, with disputes being resolved via smart contracts to guarantee eventual consistency and security [165].

7. Conclusion

In this survey, we have presented a comprehensive analysis of the Web of Agents (WoA) concept, unifying three decades of research from multi-agent systems, the Semantic Web, and modern LLM-based systems. We have argued that the field has undergone a fundamental paradigm shift: from the early vision of "semantics-in-data," which relied on formal ontologies [1], to the modern reality of "intelligence-in-model," catalyzed by the reasoning capabilities of Large Language Models (LLMs) [87]. This shift has made the WoA vision more tangible than ever before. However, this very progress has exposed a new and far more complex frontier of socio-technical risks. While past generations grappled with technological bottlenecks, the next must confront a set of deeply interconnected risks that span security, economics, identity, and governance.

The Unsolved Frontier: A Taxonomy of Risks

The creation of a truly open, decentralized, and robust Web of Agents requires moving beyond protocol design to address a landscape of systemic risks. These challenges are not independent; they form an interdependent web where a failure in one domain can trigger cascading failures in others.

A. Security and Systemic Risks.. The autonomy of agents creates novel attack surfaces. The primary threats are systemic, including **Indirect Prompt Injection (IPI)**, where malicious instructions are embedded within external data sources [153], and **Excessive Agency**, identified by OWASP as a top risk, which arises when an agent is granted excessive functionality or permissions [155]. A successful IPI attack becomes catastrophic when it can leverage an agent's excessive permissions to cause systemic damage. These vulnerabilities are not isolated; they are systemic and create a new class of "semantic exploits" that challenge the foundations of cybersecurity.

B. Economic and Viability Risks. An open WoA requires a viable economic foundation, facing risks of either failing to become self-sustaining or centralizing into a few

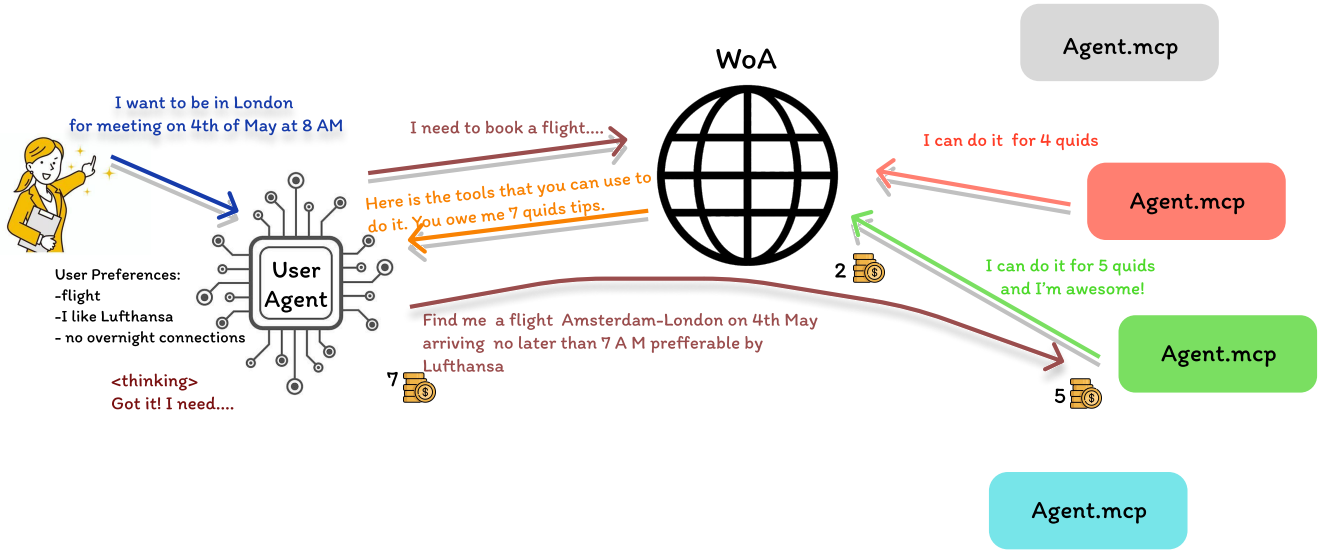

Figure 9. **An Example of an Autonomous Agent Interaction in the Web of Agents.** This figure illustrates a practical use case where a user delegates a complex task (booking a flight) to their autonomous User Agent, which then navigates the open Web of Agents to fulfill the request. The process demonstrates several key concepts and challenges discussed in the paper: **Task Delegation:** The user provides a high-level goal ("I want to be in London for a meeting") and specific preferences, offloading the cognitive work to the agent. **Service Discovery and Economic Cost:** The User Agent's first interaction is to find the necessary tools or services to book a flight. This is not a free action. It incurs a small fee paid to a directory or broker agent ("You owe me 7 quids tips"). This step shows why an economic model with micropayments is essential to compensate providers for their services and ensure the ecosystem can thrive. **Competitive Bidding:** After discovering the relevant service agents (which expose their capabilities via a protocol like MCP), the User Agent submits the specific request. Multiple service agents then respond with competing offers ("I can do it for 4 quids," "I can do it for 5 quids and I'm awesome!"). This illustrates a marketplace dynamic where agents must compete on price and performance to be selected. This entire workflow exemplifies the vision of an open agent economy, where autonomous agents discover each other, negotiate terms, and exchange value to accomplish goals on a user's behalf.

dominant platforms. This necessitates a frictionless **micropayment infrastructure** to handle the high-frequency, low-value transactions of a machine-to-machine economy, as complex agent systems can consume significant resources [139]. Without reliable signals of agent quality, the WoA is susceptible to the classic "market for lemons" problem, where low-quality agents could cause a market collapse [142]. Designing stable incentive structures, for instance through the formal modeling of **token-based economies**, is essential for fostering cooperation and ensuring long-term viability [140].

C. Trust, Identity, and Accountability Risks. Collaboration in an open, anonymous environment is impossible without trust. The WoA faces a "fundamental identity crisis" because traditional trust models break down when agents can be ephemeral, created and destroyed in milliseconds [161]. A robust "trust stack" is required, combining **Decentralized Identifiers (DIDs)** and **Verifiable Credentials (VCs)** to provide persistent, provable identities and capabilities [134]. Privacy can be preserved through techniques like **Zero-Knowledge Proofs (ZKPs)** [157]. To assess trustworthiness, these identities must be linked to **federated reputation systems** that aggregate trust ratings, using foundational algorithms like **EigenTrust** to compute global reputation scores from local peer ratings [145], [146].

D. Governance and Legal Risks. When an autonomous agent causes harm, who is responsible? The current legal landscape is unprepared, creating a "liability vacuum." This is a major risk, as agents could be empowered to agree to Terms of Service on behalf of their principals, creating legally binding agreements [156]. Establishing clear accountability is critical. New governance models are emerging that embed rules directly into the system's architecture. Frameworks like **ETHOS** propose using blockchain-based registries and Decentralized Autonomous Organizations (DAOs) for oversight [163], while new regulations like the **EU AI Act** will drive the need for auditable governance systems [164].

To conclude, the journey toward a true Web of Agents has reached a critical inflection point. The technological components are rapidly falling into place, but they are insufficient on their own. The next great research frontier is not merely technological but deeply socio-technical. Solving these interconnected risks in security, economics, trust, and governance is the central challenge for the next decade. The full realization of the WoA vision depends not on building more capable individual agents, but on collectively engineering a resilient, fair, and trustworthy *ecosystem* in which they can operate. This demands a concerted, multi-disciplinary effort from computer scientists, economists, legal scholars, and ethicists to build not just a Web of Agents, but a Web

Table 7. A SUMMARY OF SYSTEMIC RISKS FOR THE WEB OF AGENTS

Risk Domain	Specific Threats & Vulnerabilities	Required Research & Infrastructure
Security & Resilience	Indirect Prompt Injection (IPI), Excessive Agency (OWASP), Data/Model Poisoning, Cascading Failures.	Adaptive Defenses, Sandboxing, Human-in-the-Loop (HITL) for critical actions, Formal Verification.
Economic Viability	Prohibitive Transaction Costs, Centralization Risk, Incentive Misalignment, Market for Lemons.	Frictionless Micropayment Infrastructure, Stable Tokenomic Design, Robust Incentive Mechanisms.
Trust & Identity	Ephemeral Agent Identity Crisis, Sybil Attacks, Reputation Whitewashing, Lack of Verifiable Provenance.	Decentralized Identity (DID/VCs), Federated Reputation Systems (e.g., EigenTrust), Zero-Knowledge Proofs (ZKPs).
Governance & Legality	Liability Vacuum, Lack of Accountability, Regulatory Uncertainty, Emergence of "Lawless" Agents.	Sui Generis Legal Frameworks, AI Legal Personality, Decentralized Governance (DAOs), Automated Auditing.

of Trust.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Scientific American*, vol. 284, no. 5, pp. 34–43, May 2001.

[2] Google, "Agent-to-Agent (A2A) protocol specification," 2025, accessed: 2025-07-12. [Online]. Available: https://a2aproject.github.io/A2A/latest/

[3] Anthropic, "Model Context Protocol (MCP) specification," November 2024, accessed: 2025-07-12. [Online]. Available: https://modelcontextprotocol.io/introduction

[4] Foundation for Intelligent Physical Agents, "FIPA ACL Message Structure Specification," FIPA Standard SC00061G, December 2002, accessed: 2025-07-12. [Online]. Available: http://www.fipa.org/specs/fipa00061/SC00061G.html

[5] W3C OWL Working Group, "OWL 2 web ontology language document overview (second edition)," W3C Recommendation, 11 December 2012, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/2012/REC-owl2-overview-20121211/

[6] OpenAI, "Introducing GPT-4.5," OpenAI Blog, February 27, 2025, accessed: 2025-07-12. [Online]. Available: https://openai.com/index/introducing-gpt-4-5/

[7] Google DeepMind, "Gemini 2.5: Our newest Gemini model with thinking," Google DeepMind Blog, March 28, 2025, accessed: 2025-07-12. [Online]. Available: https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/

[8] Anthropic, "Claude 3.7 Sonnet and Claude Code," Anthropic Blog, February 24, 2025, accessed: 2025-07-12. [Online]. Available: https://www.anthropic.com/news/claude-3-7-sonnet

[9] xAI, "Grok 3 Beta — The Age of Reasoning Agents," xAI Blog, February 19, 2025, accessed: 2025-07-12. [Online]. Available: https://x.ai/news/grok-3

[10] Amazon Web Services, "Mistral Large 2 foundation model now available in Amazon Bedrock," AWS News, July 24, 2024, accessed: 2025-07-12. [Online]. Available: https://aws.amazon.com/about-aws/whats-new/2024/07/mistral-large-2-foundation-model-amazon-bedrock/

[11] Alibaba DAMO Academy, "Alibaba unveils Qwen 3: A family of hybrid AI reasoning models," April 28, 2025, accessed: 2025-07-12. [Online]. Available: https://www.alibabacloud.com/blog/alibaba-introduces-qwen3-setting-new-benchmark-in-open-source-ai-with-hybrid-reasoning_602192

[12] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus, "Emergent abilities of large language models," *Transactions on Machine Learning Research*, August 2022, accessed: 2025-07-12. [Online]. Available: https://openreview.net/forum?id=yzkSU5zdwD

[13] P. Shojaee, I. Mirzadeh, K. Alizadeh, M. Horton, S. Bengio, and M. Farajtabar, "The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity," arXiv preprint arXiv:2506.06941, June 2025, also available at: https://machinelearning.apple.com/research/illusion-of-thinking. [Online]. Available: https://arxiv.org/abs/2506.06941

[14] Elsevier, "Scopus: Abstract and citation database," https://www.scopus.com, November 2004, accessed: 2025-07-12.

[15] Y. Du, S. Li, A. Torralba, J. B. Tenenbaum, and I. Mordatch, "Improving factuality and reasoning in language models through multiagent debate," in *Proceedings of the 41st International Conference on Machine Learning*, Vienna, Austria. PMLR 235, 2024, accessed: 2025-07-12. [Online]. Available: https://openreview.net/pdf?id=zj7YuTE4t8

[16] S. P. Mousavi Davoudi, A. Gholami Davodi, A. Amiri-Margavi, and M. Jafari, "Collective Reasoning Among LLMs: A Framework for Answer Validation Without Ground Truth," arXiv preprint arXiv:2502.20758, 2025, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2502.20758

[17] B. Matthews, "Semantic web technologies," *E-Learning and Digital Media*, vol. 6, no. 6, pp. 725–731, 2005, accessed: 2025-07-12. [Online]. Available: https://www.researchgate.net/publication/30408878_Semantic_Web_Technologies

[18] N. Malik and S. K. Malik, "Semantic web as the next generation smart web," *Journal of Emerging Technologies and Innovative Research (JETIR)*, vol. 5, no. 6, pp. 580–594, 2018, accessed: 2025-07-12. [Online]. Available: https://www.jetir.org/papers/JETIR1806326.pdf

[19] P. Mika and J. Akkermans, "Towards a new synthesis of ontology technology and knowledge management," *The Knowledge Engineering Review*, vol. 19, no. 4, pp. 317–345, 2004, accessed: 2025-07-12. [Online]. Available: https://doi.org/10.1017/S0269888905000305

[20] G. Weiss, Ed., *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA: MIT Press, 1999.

[21] D. Brickley and R. V. Guha, "RDF vocabulary description language 1.0: RDF schema," W3C Recommendation, February 2004, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/2004/REC-rdf-schema-20040210/

[22] D. L. McGuinness and F. van Harmelen, "OWL web ontology language overview," W3C Recommendation, February 2004, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/2004/REC-owl-features-20040210/

[23] A. Ciortea, S. Mayer, F. Gandon, O. Boissier, A. Ricci, and A. Zimmermann, "A decade in hindsight: The missing bridge between multi-agent systems and the world wide web," in *Proceedings of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. Montreal, Canada: International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1659–1663, accessed: 2025-07-12. [Online]. Available: https://hal.science/emse-02070625/

[24] R. Sapkota, K. I. Roumeliotis, and M. Karkee, "AI agents vs. agentic AI: A conceptual taxonomy, applications and challenges," arXiv preprint arXiv:2505.10468, 2025, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2505.10468

[25] M. A. Ferrag, N. Tihanyi, and M. Debbah, "From LLM reasoning to autonomous AI agents: A comprehensive review," arXiv preprint arXiv:2504.19678, 2025, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2504.19678

[26] L. Wang, C. Ma, X. Feng, Z. Zhang, H. Yang, J. Zhang, Z. Chen, J. Tang, X. Chen, Y. Lin, W. X. Zhao, Z. Wei, and J.-R. Wen, "A survey on large language model based autonomous agents," arXiv preprint arXiv:2308.11432, 2023, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2308.11432

[27] A. Ehtesham, A. Singh, G. K. Gupta, and S. Kumar, "A survey of agent interoperability protocols: Model Context Protocol (MCP), Agent Communication Protocol (ACP), Agent-to-Agent Protocol (A2A), and Agent Network Protocol (ANP)," arXiv preprint arXiv:2505.02279, 2025, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2505.02279

[28] Agent Network Protocol Contributors, "Agent Network Protocol Technical White Paper: Towards an Open Internet of Agents," 2025, accessed: 2025-07-12. [Online]. Available: https://agentnetworkprotocol.com/en/specs/01-agentnetworkprotocol-technical-white-paper/

[29] A. S. Rao and M. P. Georgeff, "BDI agents: From theory to practice," in *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS)*. AAAI Press, 1995, pp. 312–319.

[30] T. Finin, R. Fritzson, D. McKay, and R. McEntire, "KQML as an agent communication language," in *Proceedings of the Third International Conference on Information and Knowledge Management (CIKM '94)*. ACM Press, 1994, pp. 456–463, accessed: 2025-07-12. [Online]. Available: https://dl.acm.org/doi/10.1145/191246.191322

[31] S. Murugesan, "Intelligent agents on the Internet and Web," in *Proceedings of TENCON '98. IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control*. IEEE, 1998, pp. 97–102, accessed: 2025-07-12. [Online]. Available: https://ieeexplore.ieee.org/document/797088

[32] H. S. Nwana, "Software agents: An overview," *The Knowledge Engineering Review*, vol. 11, no. 3, pp. 205–244, 1996, accessed: 2025-07-12. [Online]. Available: https://doi.org/10.1017/S026988890000789X

[33] D. B. Lange and M. Oshima, *Programming and Deploying Java Mobile Agents with Aglets*. Boston, MA: Addison-Wesley, 1998.

[34] F. Bellifemine, A. Poggi, and G. Rimassa, "JADE: A FIPA-compliant agent framework," in *Proceedings of the 4th International Conference on The Practical Application of Intelligent Agents and Multi-Agent Systems (PAAM'99)*, London, UK, 1999, pp. 97–108, accessed: 2025-07-12. [Online]. Available: https://jade.tilab.com/papers/PAAM99.pdf

[35] Y. Yao, "Web intelligence: New frontiers of exploration," in *Proceedings of the 2005 International Conference on Active Media Technology (AMT 2005)*. IEEE, 2005, pp. 3–8.

[36] IEEE/WIC, "The 23rd IEEE/WIC international conference on web intelligence and intelligent agent technology (WI-IAT'24)," in *Proceedings of the 2024 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. Bangkok, Thailand: IEEE, 2024, accessed: 2025-07-12. [Online]. Available: https://dblp.org/db/conf/webi/webi2024.html

[37] D. A. Domingos, B. M. Faria, A. M. Z-Flores, S. Isotani, I. I. Bittencourt, and J. Cascalho, "Web intelligence journal in perspective: An analysis of its two decades trajectory," *arXiv preprint arXiv:2405.05129*, 2024, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2405.05129

[38] Significant-Gravitas, "AutoGPT: An autonomous GPT agent," GitHub repository, 2023, accessed: 2025-07-12. [Online]. Available: https://github.com/Significant-Gravitas/AutoGPT

[39] F. Manola and E. Miller, "RDF primer," W3C Recommendation, February 2004, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/2004/REC-rdf-primer-20040210/

[40] I. Horrocks, P. F. Patel-Schneider, and F. van Harmelen, "From SHIQ and RDF to OWL: The making of a web ontology language," *Journal of Web Semantics*, vol. 1, no. 1, pp. 7–26, 2003.

[41] D. Brickley and R. V. Guha, "RDF schema 1.1," W3C Recommendation, February 2014, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/2014/REC-rdf-schema-20140225/

[42] D. Fensel, *Ontologies: A silver bullet for knowledge management and electronic commerce*. Berlin, Heidelberg: Springer-Verlag, 2001.

[43] Z. Huang, A. Eliëns, A. van Ballegooij, and P. de Bra, "A taxonomy of web agents," in *Proceedings of the 11th International Workshop on Database and Expert Systems Applications (DEXA'00)*, A. M. Toja, Ed. Los Alamitos, CA, USA: IEEE Computer Society, September 2000, pp. 765–769.

[44] N. Shadbolt, T. Berners-Lee, and W. Hall, "The semantic web revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, 2006.

[45] T. Heindel and I. Weber, "Incentive alignment of business processes," in *Business Process Management - 18th International Conference, BPM 2020, Seville, Spain, September 13-18, 2020, Proceedings*, ser. Lecture Notes in Computer Science, vol. 12168. Cham: Springer, 2020, pp. 114–131.

[46] E. Simperl, R. Cuel, and M. Stein, *Incentive-Centric Semantic Web Application Engineering*, ser. Synthesis Lectures on the Semantic Web: Theory and Technology. San Rafael, CA: Morgan & Claypool Publishers, 2013.

[47] J. Moilanen, M. Niinioja, and M. Seppänen, *API Economy 101: Changes Your Business*. Norderstedt: Books on Demand, 2019.

[48] K. Siorpaes and E. Simperl, Eds., *Proceedings of the 1st Workshop on Incentives for the Semantic Web (INSEMTIVE 2008)* at ISWC 2008, Karlsruhe, Germany, October 26, 2008, accessed: 2025-07-12. [Online]. Available: http://km.aifb.kit.edu/ws/insemtive2008/

[49] Facebook, Inc., "The open graph protocol," 2010, accessed: 2025-07-12. [Online]. Available: https://ogp.me/

[50] A. Hogan, "The semantic web: Two decades on," *Semantic Web*, vol. 11, no. 1, pp. 169–185, 2020.

[51] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "DBpedia – A crystallization point for the web of data," *Journal of Web Semantics*, vol. 7, no. 3, pp. 154–165, 2009.

[52] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledge base," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[53] A. Iliadis, A. Acker, W. Stevens, and S. B. Kavakli, "One schema to rule them all: How Schema.org models the world of search," *Journal of the Association for Information Science and Technology*, vol. 76, no. 2, pp. 460–523, 2025.

[54] R. V. Guha, D. Brickley, and S. Macbeth, "Schema.org: Evolution of structured data on the web," *Communications of the ACM*, vol. 59, no. 2, pp. 44–51, 2016.

[55] M. Wooldridge, *An introduction to MultiAgent systems*, 2nd ed. Chichester, UK: John Wiley & Sons, 2009.

[56] M. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995.

[57] V. R. Lesser, "A retrospective view of FA/C distributed problem solving," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, no. 6, pp. 1347–1362, Dec. 1991.

[58] M. N. Huhns, "Distributed artificial intelligence," in *Distributed artificial intelligence*, L. Gasser and M. N. Huhns, Eds. Morgan Kaufmann, 1989, pp. 1–23.

[59] L. D. Erman, F. Hayes-Roth, V. R. Lesser, and D. R. Reddy, "The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty," *ACM Computing Surveys*, vol. 12, no. 2, pp. 213–253, June 1980.

[60] D. D. Corkill, "A framework for organizational self-design in distributed problem solving networks," Ph.D. dissertation, Department of Computer and Information Science, University of Massachusetts, Amherst, Technical Report 82-33, 1982.

[61] S. Poslad, "History of FIPA," Foundation for Intelligent Physical Agents, Queen Mary, University of London, last updated 20 December 2005, accessed: 2025-07-12. [Online]. Available: http://www.fipa.org/subgroups/ROFS-SG-docs/History-of-FIPA.htm

[62] S. Poslad, "Specifying protocols for multi-agent systems interaction," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 2, no. 4, Article 15, pp. 15:1–15:24, November 2007.

[63] V. Mascardi, J. Hendler, and L. Papaleo, "Semantic web and declarative agent languages and technologies: Current and future trends," in *Declarative Agent Languages and Technologies X*, ser. Lecture Notes in Computer Science, M. Baldoni, L. Dennis, V. Mascardi, and W. Vasconcelos, Eds. Berlin, Heidelberg: Springer, 2013, vol. 7784, pp. 197–202.

[64] T. Manev and S. Filiposka, "Semantic aware multi-agent system advantages," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 3, no. 1, pp. 1–12, February 2014.

[65] S. Coppens, R. Verborgh, E. Mannens, and R. Van de Walle, "Self-sustaining platforms: A semantic workflow engine," in *Proceedings of the 4th International Workshop on Consuming Linked Data*, ser. CEUR Workshop Proceedings, vol. 1034. Sydney, Australia: CEUR-WS.org, October 2013, pp. 1–8, accessed: 2025-07-12. [Online]. Available: http://ceur-ws.org/Vol-1034/CoppensEtAl_COLD2013.pdf

[66] Y. Labrou, T. Finin, and Y. Peng, "Agent communication languages: The current landscape," *IEEE Intelligent Systems*, vol. 14, no. 2, pp. 45–52, Mar./Apr. 1999, doi: 10.1109/5254.757631. [Online]. Available: https://doi.org/10.1109/5254.757631

[67] M. Laclavík, Z. Balogh, M. Babík, and L. Hluchý, "AgentOWL: Semantic knowledge model and agent architecture," *Computing and Informatics*, vol. 25, no. 5, pp. 421–439, 2006.

[68] V. Charpenay, M. Baldoni, A. Ciortea, S. Cranefield, J. Padget, and M. P. Singh, Eds. Cham: Springer Nature Switzerland, 2023, pp. 53–71, accessed: 2025-07-12. [Online]. Available: https://doi.org/10.1007/978-3-031-49133-7_4

[69] O. Shafiq, Y. Ding, and D. Fensel, "Bridging multi agent systems and web services: Towards interoperability between software agents and semantic web services," in *Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC '06)*. Hong Kong, China: IEEE Computer Society, October 2006, pp. 85–96.

[70] F. Cavallaro, F. Giunchiglia, and I. Liccardi, "A middleware architecture for semantic web services," in *The Semantic Web: Research and Applications*, ser. Lecture Notes in Computer Science, C. Bussler, J. Davies, D. Fensel, and R. Studer, Eds., vol. 3053. Berlin, Heidelberg: Springer, 2004, pp. 67–81.

[71] M. Pasha, S. Rehman, A. Ali, H. F. Ahmad, and H. Suguri, "Middleware between OWL and FIPA ontologies in the semantic grid environment," in *Proceedings of the 2006 International Conference on Semantic Web and Web Services (SWWS '06)*. Las Vegas, NV, USA: CSREA Press, June 2006, pp. 30–35, accessed: 2025-07-12. [Online]. Available: https://dblp.org/rec/conf/swws/PashaRAAS06

[72] D. B. Acharya, K. Kuppan, and D. B. Ashwin, "Agentic AI: A comprehensive survey of autonomous, intelligent, and goal-directed AI systems," *IEEE Access*, vol. 12, pp. 27 338–27 357, 2024.

[73] J. Schneider, "Generative to agentic AI: Survey, conceptualization, and challenges," *arXiv preprint arXiv:2504.18875*, 2025. [Online]. Available: https://arxiv.org/abs/2504.18875

[74] R. Sharma, M. de Vos, P. Chari, R. Raskar, and A. Kermarrec, "Collaborative agentic AI needs interoperability across ecosystems," *arXiv preprint arXiv:2505.21550*, 2025. [Online]. Available: https://arxiv.org/abs/2505.21550

[75] N. R. Jennings, K. Sycara, and M. Wooldridge, "A roadmap of agent research and development," *Autonomous Agents and Multi-Agent Systems*, vol. 1, no. 1, pp. 7–38, 1998, doi: 10.1023/A:1010090405266. [Online]. Available: https://doi.org/10.1023/A:1010090405266

[76] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28,573–28,593, 2018, doi: 10.1109/ACCESS.2018.2835227. [Online]. Available: https://ieeexplore.ieee.org/document/8352646

[77] N. Shadbolt, W. Hall, and T. Berners-Lee, "The semantic web revisited," *IEEE Intelligent Systems*, vol. 21, no. 3, pp. 96–101, 2006, doi: 10.1109/MIS.2006.62. [Online]. Available: https://doi.org/10.1109/MIS.2006.62

[78] J. Dale and A. Johnson, "Rational agents for decentralized environments," in *Proceedings of the Grid-Interop forum 2007*, Seattle, WA, USA, 2007, accessed: 2025-07-12. [Online]. Available: https://gridwiseac.org/pdfs/forum_papers/135_paper_final3.pdf

[79] H. Coelho, "Future challenges for autonomous systems," in *Artificial Intelligence: An International Perspective*, M. Bramer, Ed., ser. Lecture Notes in Computer Science, vol. 5640. Berlin, Heidelberg: Springer, 2009, pp. 39–52.

[80] L. J. Moya, "Towards a taxonomy of agents and multi-agent systems," in *Proceedings of the 2007 Spring Simulation Multiconference*, San Diego, CA, USA: Society for Computer Simulation International, 2007, pp. 27–34.

[81] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing reasoning and acting in language models," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023. [Online]. Available: https://arxiv.org/abs/2210.03629

[82] D. Bai, I. Singh, D. Traum, and J. Thomason, "TwoStep: Multi-agent task planning using classical planners and large language models," *arXiv preprint arXiv:2403.17246*, 2024. [Online]. Available: https://arxiv.org/abs/2403.17246

[83] J. Alejandre, A. Rincon, and M. Lopez-Sanchez, "Hatp: Hierarchical agent-based task planner," in *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '18)*, Stockholm, Sweden: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 1823–1825, accessed: 2025-07-12. [Online]. Available: http://www.ifaamas.org/Proceedings/aamas2018/pdfs/p1823.pdf

[84] K. Bauters, K. McAreavey, J. Hong, Y. Chen, W. Liu, L. Godo, and C. Sierra, "Probabilistic planning in agentspeak using the pomdp framework," in *Combinations of Intelligent Methods and Applications*, ser. Smart Innovation, Systems and Technologies, I. Hatzilygeroudis, V. Palade, and J. Prentzas, Eds., Cham: Springer International Publishing, 2016, vol. 46, pp. 19–37.

[85] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. [Online]. Available: https://arxiv.org/abs/2305.10601

[86] S. Zhang, B. Peng, X. Zhao, B. Hu, Y. Zhu, Y. Zeng, and X. Hu, "LLaSA: Large language and e-commerce shopping assistant," *arXiv preprint arXiv:2408.02006*, 2024. [Online]. Available: https://arxiv.org/abs/2408.02006

[87] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 1877–1901, accessed: 2025-07-12. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html

[88] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg, H. Nori, H. Palangi, M. T. Ribeiro, and Y. Zhang, "Sparks of artificial general intelligence: Early experiments with GPT-4," *arXiv preprint arXiv:2303.12712*, 2023. [Online]. Available: https://arxiv.org/abs/2303.12712

[89] Google, "Agent2Agent (A2A) protocol repository," https://github.com/google/A2A, 2025, accessed: 2025-07-12.

[90] T. Schick, J. Dwivedi-Yu, R. Dessi, R. Raileanu, M. Cettolo, L. Sestagalli, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," in *Advances in Neural Information Processing Systems*, vol. 36, 2023, pp. 68 539–68 551. [Online]. Available: https://arxiv.org/abs/2302.04761

[91] P. Lewis, E. Pérez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, S. Welleck, M. Komeili, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, 2020, pp. 9459–9474, accessed: 2025-07-12. [Online]. Available: https://arxiv.org/abs/2005.11401

[92] OpenAI, "Introducing ChatGPT plugins," OpenAI Blog, March 2023, accessed: 2025-07-12. [Online]. Available: https://openai.com/blog/chatgpt-plugins

[93] LangChain Authors, "LangChain: A framework for building LLM applications with tooling," https://python.langchain.com/, 2023, accessed: 2025-07-12.

[94] langchain-ai, "LangChain: A framework for developing applications powered by large language models," 2023, accessed: 2025-07-12. [Online]. Available: https://github.com/langchain-ai/langchain

[95] B. Burns, "Borg, the predecessor to Kubernetes," *Kubernetes Blog*, Apr. 2015, accessed: 2025-07-12. [Online]. Available: https://kubernetes.io/blog/2015/04/borg-predecessor-to-kubernetes/

[96] Google Cloud, "AI and ML orchestration on GKE," *Google Kubernetes Engine Documentation*, accessed: 2025-07-12. [Online]. Available: https://cloud.google.com/kubernetes-engine/docs/integrations/ai-infra

[97] A. Gupta, "Agentic AI on Kubernetes: Advanced orchestration, deployment, and scaling strategies for autonomous AI systems," *Collabnix*, May 2024, accessed: 2025-07-12. [Online]. Available: https://collabnix.com/agentic-ai-on-kubernetes-advanced-orchestration-deployment-and-scaling-strategies-for-autonomous-ai-systems/

[98] A. Verma, L. Pedrosa, M. R. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes, "Large-scale cluster management at Google with Borg," in *Proceedings of the Tenth European Conference on Computer Systems (EuroSys '15)*, New York, NY, USA: ACM, 2015, pp. 1–17, doi: 10.1145/2741948.2741964. [Online]. Available: https://doi.org/10.1145/2741948.2741964

[99] P. F. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, "Deep reinforcement learning from human preferences," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, 2017, pp. 4299–4307, accessed: 2025-07-12. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/d5e2c0adad503c91f91df240d0cd4e49-Abstract.html

[100] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, Eds., vol. 15. PMLR, 2011, pp. 627–635, accessed: 2025-07-12. [Online]. Available: https://proceedings.mlr.press/v15/ross11a.html

[101] GDELT Project, "The GDELT project: Global database of events, language, and tone," GDELT Website, January 2011, accessed: 2025-07-12. [Online]. Available: https://www.gdeltproject.org/

[102] Google Trends, "Google Trends," https://trends.google.com/trends/, n.d., accessed: 2025-07-12.

[103] Microsoft, "NLWeb: Bringing conversational interfaces directly to the web," https://github.com/microsoft/NLWeb, 2025, accessed: 2025-07-12.

[104] Docker Inc., "The model context protocol: An emerging standard for AI agent-tool interactions," White Paper, Docker Inc., 2025, accessed: 2025-07-12. [Online]. Available: https://www.docker.com/resources/the-model-context-protocol-white-paper/

[105] V. Charpenay, M. Baldoni, S. Cranefield, M. Dastani, L. Dennis, P. Noriega, J. Padget, M. P. Singh, L. van der Torre, and S. Villata, "Governing agents on the web," in *Proceedings of the COINE Workshop*, London, UK, 2023, accessed: 2025-07-14. [Online]. Available: https://coin-workshop.github.io/coine-2023-london/Papers/Paper-10.pdf

[106] CrewAI Contributors, "CrewAI: A multi-agent orchestration toolkit," 2024, accessed: 2025-07-12. [Online]. Available: https://github.com/crewAIInc/crewAI

[107] SaM-92, "`mcp_autogen_sse_stdio`: AutoGen MCP streaming I/O adapter," 2025, accessed: 2025-07-12. [Online]. Available: https://github.com/SaM-92/mcp_autogen_sse_stdio

[108] C. Liguori, "Introducing Strands Agents, an open source AI agents SDK," *AWS Open Source Blog*, May 16, 2025, accessed: 2025-07-12. [Online]. Available: https://aws.amazon.com/ru/blogs/opensource/introducing-strands-agents-an-open-source-ai-agents-sdk/

[109] Microsoft, "Semantic Kernel: Orchestration and plugins for AI," https://github.com/microsoft/semantic-kernel, 2023, accessed: 2025-07-12.

[110] Microsoft, "Semantic Kernel: Agent-to-Agent examples," 2024, accessed: 2025-07-12. [Online]. Available: https://github.com/microsoft/semantic-kernel/tree/main

[111] LlamaIndex Development Team, "LlamaIndex: Build knowledge assistants over your enterprise data," https://www.llamaindex.ai/, accessed: 2025-07-12.

[112] TransformerOptimus, "SuperAGI: A dev-first open-source autonomous AI agent framework," https://github.com/TransformerOptimus/SuperAGI, 2023, accessed: 2025-07-12.

[113] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, C. Zhang, J. Wang, Z. Wang, S. K. S. Yau, Z. Lin, L. Zhou, C. Ran, L. Xiao, C. Wu, and J. Schmidhuber, "MetaGPT: Meta programming for a multi-agent collaborative framework," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. [Online]. Available: https://arxiv.org/abs/2308.00352

[114] Microsoft Research, "Autogen: Enabling next-gen llm applications via multi-agent conversation," GitHub repository https://github.com/microsoft/autogen, 2023, accessed: 2025-07-12.

[115] CrewAI Contributors, "Crewai: Framework for orchestrating role-playing, autonomous AI agents," GitHub repository https://github.com/joaomdmoura/crewAI, 2023, accessed: 2025-07-12.

[116] OpenAI, "Assistants API," *OpenAI API Documentation*, 2023, accessed: 2025-07-12. [Online]. Available: https://platform.openai.com/docs/assistants/overview

[117] MetaGPT Contributors, "Metagpt: Multi-agent framework," GitHub repository https://github.com/geekan/MetaGPT, 2023, accessed: 2025-07-12.

[118] Google Cloud, "Overview of Agent Builder," *Vertex AI Documentation*, 2024, accessed: 2025-07-12. [Online]. Available: https://cloud.google.com/vertex-ai/docs/generative-ai/agent-builder/overview

[119] Amazon Web Services, "Amazon bedrock agents," https://aws.amazon.com/bedrock/agents/, 2023, accessed: 2025-07-12.

[120] Microsoft Azure, "Azure AI Studio," *Microsoft Azure*, 2024, accessed: 2025-07-12. [Online]. Available: https://azure.microsoft.com/en-us/products/ai-studio

[121] Anthropic, "Build with Claude," *Anthropic Documentation*, 2024, accessed: 2025-07-12. [Online]. Available: https://docs.anthropic.com/en/docs/using-the-computer

[122] IBM, "IBM watsonx Orchestrate," *IBM*, 2023, accessed: 2025-07-12. [Online]. Available: https://www.ibm.com/products/watsonx-orchestrate

[123] J. Song, Z. Ashktorab, and T. W. Malone, "Togedule: Scheduling meetings with large language models and adaptive representations of group availability," *arXiv preprint arXiv:2505.01000*, 2025. [Online]. Available: https://arxiv.org/abs/2505.01000

[124] Y. Li, S. Ma, X. Wang, S. Huang, C. Jiang, H.-T. Zheng, P. Xie, F. Huang, and Y. Jiang, "EcomGPT: Instruction-tuning large language models with chain-of-task tasks for e-commerce," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 17, 2024, pp. 18582–18590. [Online]. Available: https://doi.org/10.1609/aaai.v38i17.29820

[125] W3C Spatial Data on the Web Working Group, "Time ontology in OWL," W3C Recommendation, October 2017, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/owl-time/

[126] TransformerOptimus, "SuperAGI: A dev-first open source autonomous AI agent framework," https://github.com/TransformerOptimus/SuperAGI, 2023, accessed: 2025-07-12.

[127] Microsoft, "Magentic-UI: A research prototype of a human-centered interface powered by a multi-agent system," https://github.com/microsoft/magentic-ui, 2025, accessed: 2025-07-12.

[128] Amazon Web Services, "AI foundation model marketplace - Amazon Bedrock Marketplace," 2024, accessed: 2025-07-12. [Online]. Available: https://aws.amazon.com/bedrock/marketplace/

[129] OpenAI, "Introducing the GPT Store," January 2024, accessed: 2025-07-12. [Online]. Available: https://openai.com/index/introducing-the-gpt-store/

[130] Google Cloud, "Vertex AI Agent Builder overview," 2024, accessed: 2025-07-12. [Online]. Available: https://cloud.google.com/vertex-ai/generative-ai/docs/agent-builder/overview

[131] G. Rehm, D. Galanis, P. Labropoulou, S. Piperidis, M. Welß, R. Usbeck, J. Köhler, M. Deligiannis, K. Gkirtzou, J. Fischer, C. Chiarcos, N. Feldhus, J. Moreno-Schneider, F. Kintzel, E. Montiel, V. Rodríguez Doncel, J. P. McCrae, D. Laqua, I. P. Theile, C. Dittmar, K. Bontcheva, I. Roberts, A. Vasiljevs, and A. Lagzdiņš, "Towards an interoperable ecosystem of AI and LT platforms: A roadmap for the implementation of different levels of interoperability," *arXiv preprint arXiv:2004.08355*, 2020. [Online]. Available: https://arxiv.org/pdf/2004.08355

[132] N. Yekollu, R. Jain, and S. G. Patil, "Agent marketplace," 2024. [Online]. Available: https://gorilla.cs.berkeley.edu/blogs/11_agent_marketplace.html

[133] S. Zaragocin, K. P. Krishnan, B. T. K. Tan, and J. M. Zamen, "ETHOS: Exploring multi-agent reinforcement learning for unbiased federated learning," *arXiv preprint arXiv:2412.17114*, 2024. [Online]. Available: https://arxiv.org/pdf/2412.17114

[134] World Wide Web Consortium, "Verifiable Credentials Data Model v2.0," World Wide Web Consortium, W3C Recommendation, May 2025, accessed: 2025-07-12. [Online]. Available: https://www.w3.org/TR/vc-data-model-2.0/

[135] K. Huang, V. S. Narajala, J. Yeoh, J. Ross, R. Raskar, Y. Harkati, J. Huang, I. Habler, and C. Hughes, "A Novel Zero-Trust Identity Framework for Agentic AI: Decentralized Authentication and Fine-Grained Access Control," *arXiv preprint arXiv:2505.19301*, May 2025. [Online]. Available: https://arxiv.org/abs/2505.19301

[136] A. Clemm, L. Ciavaglia, L. Z. Granville, and J. Tantsura, "Intent-based networking - concepts and definitions," Internet Research Task Force (IRTF), RFC 9315, October 2022. [Online]. Available: https://www.ietf.org/rfc/rfc9315.html

[137] T. Liu, G. S. Ramachandran, and R. Jurdak, "Post-quantum cryptography for internet of things: A survey on performance and optimization," *arXiv preprint arXiv:2401.17538*, 2024. [Online]. Available: https://arxiv.org/html/2401.17538v1/

[138] R. Noothigattu, S. Gaikwad, E. Awad, S. Dsouza, I. Rahwan, P. Ravikumar, and A. D. Procaccia, "A voting-based system for ethical decision making," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 1587–1594. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/11512

[139] Anthropic, "How we built our multi-agent research system," Anthropic Engineering Blog, June 2025, accessed: 2025-07-12. [Online]. Available: https://www.anthropic.com/engineering/built-multi-agent-research-system

[140] R. Sadykhov, G. Goodell, D. de Montigny, M. Schoernig, and P. Treleaven, "Decentralized token economy theory (DeTEcT): token pricing, stability and governance for token economies," arXiv preprint arXiv:2309.12330, 2023. [Online]. Available: https://arxiv.org/pdf/2309.12330

[141] D. Yermack, "Is bitcoin a real currency? An economic appraisal," in *Handbook of Digital Currency*, D. K. C. Lee, Ed. Amsterdam: Elsevier, 2015, pp. 31–43.

[142] G. A. Akerlof, "The market for "lemons": Quality uncertainty and the market mechanism," *The Quarterly Journal of Economics*, vol. 84, no. 3, pp. 488–500, 1970.

[143] J. Sabater and C. Sierra, "Review on computational trust and reputation models," *Artificial Intelligence Review*, vol. 24, no. 1, pp. 33–60, 2005. [Online]. Available: https://doi.org/10.1007/s10462-004-0041-5

[144] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007. [Online]. Available: https://doi.org/10.1016/j.dss.2005.05.019

[145] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt, "An integrated trust and reputation model for open multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 2, pp. 119–154, 2006. [Online]. Available: https://doi.org/10.1007/s10458-005-6825-4

[146] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust algorithm for reputation management in P2P networks," in *Proceedings of the 12th International Conference on World Wide Web (WWW '03)*. New York, NY, USA: ACM, 2003, pp. 640–651. [Online]. Available: https://doi.org/10.1145/775152.775242

[147] Ö. Doğan and H. Karacan, "A blockchain-based e-commerce reputation system built with verifiable credentials," *IEEE Access*, vol. 11, pp. 49227–49238, 2023. [Online]. Available: https://doi.org/10.1109/ACCESS.2023.3274707

[148] R. Jurca and B. Faltings, "Truthful feedback for sanctioning reputation mechanisms," in *Proceedings of the 6th ACM Conference on Electronic Commerce (EC '05)*. New York, NY, USA: ACM, 2005, pp. 190–199.

[149] E. Friedman and P. Resnick, "The social cost of cheap pseudonyms," *Journal of Economics & Management Strategy*, vol. 10, no. 2, pp. 173–199, 2001. [Online]. Available: https://smg.media.mit.edu/library/FriedmanResnick.pseudonyms.pdf (accessed: 2025-07-12).

[150] A. Battah, Y. Iraqi, and E. Damiani, "Blockchain-based reputation systems: Implementation challenges and mitigation," *Electronics*, vol. 10, no. 3, p. 289, 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/3/289 (accessed: 2025-07-12).

[151] M. Masse, *REST API Design Rulebook: Designing Consistent Web Services*. O'Reilly Media, 2016.

[152] S. Heshmatisafa and M. Seppänen, "Exploring API-driven business models: Lessons learned from Amadeus's digital transformation," *Digital Business*, vol. 3, no. 1, p. 100055, 2023. [Online]. Available: https://doi.org/10.1016/j.digbus.2023.100055

[153] Y. Liu, G. Deng, Y. Li, K. Wang, T. Zhang, Y. Liu, H. Wang, Y. Zheng, and Y. Liu, "Prompt injection attack against LLM-integrated applications," arXiv preprint arXiv:2306.05499, 2023. [Online]. Available: https://arxiv.org/abs/2306.05499 (accessed: 2025-07-12).

[154] P. W. Koh and P. Liang, "Understanding black-box predictions via influence functions," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. PMLR, 2017, pp. 1885–1894, accessed: 2025-07-12. [Online]. Available: http://proceedings.mlr.press/v70/koh17a/koh17a.pdf

[155] OWASP Foundation, "OWASP Top 10 for Large Language Model Applications," 2024, version 1.1, accessed: 2025-07-12. [Online]. Available: https://owasp.org/www-project-top-10-for-large-language-model-applications/

[156] National Conference of Commissioners on Uniform State Laws, "Uniform Electronic Transactions Act," 1999, accessed: 2025-07-12. [Online]. Available: http://euro.ecom.cmu.edu/program/law/08-732/Transactions/ueta.pdf

[157] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on Computing*, vol. 18, no. 1, pp. 186–208, 1989.

[158] A. Nait Cherif, Y. Achir, M. Youssfi, M. Elgarej, and O. Bouattane, "Zero-Knowledge Proofs and OAuth 2.0 for Anonymity and Security in Distributed Systems," in *E3S Web of Conferences*, vol. 469, 2023, p. 00085. [Online]. Available: https://doi.org/10.1051/e3sconf/202346900085

[159] A. Gabizon, Z. J. Williamson, and O. Ciobotaru, "PLONK: Permutations over Lagrange-bases for Oecumenical Noninteractive Arguments of Knowledge," Cryptology ePrint Archive, Report 2019/953, 2019. [Online]. Available: https://eprint.iacr.org/2019/953

[160] A. Lior, "Insuring AI: The Role of Insurance in Artificial Intelligence Regulation," *Harvard Journal of Law & Technology*, vol. 35, no. 2, pp. 469–529, 2022, accessed: 2025-07-12. [Online]. Available: https://jolt.law.harvard.edu/assets/articlePDFs/v35/2.-Lior-Insuring-AI.pdf

[161] K. C. Toth, "Agent-Based Digital Identity Architecture," in *Proceedings of the Pacific Northwest Software Quality Conference (PNSQC)*, 2019, pp. 1–12, accessed: 2025-07-12. [Online]. Available: https://www.pnsqc.org/docs/Toth_Agent-BasedDigital_Final.pdf

[162] European Commission, "European Digital Identity Wallet: Security and Privacy Features," European Commission, 2024, accessed: 2025-07-12. [Online]. Available: https://ec.europa.eu/digital-building-blocks/sites/display/EUDIGITALIDENTITYWALLET/Security+and+Privacy

[163] T. J. Chaffer, J. Goldston, B. Okusanya, and Gemach D.A.T.A. I, "On the ETHOS of AI Agents: An Ethical Technology and Holistic Oversight System," arXiv preprint arXiv:2412.17114, 2024.

[164] European Parliament and Council of the European Union, "Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act)," Official Journal of the European Union, pp. 1–98, Jul. 2024. Regulation (EU) 2024/1689; Accessed: 2025-07-12. [Online]. Available: http://data.europa.eu/eli/reg/2024/1689/oj/eng

[165] Z. Hong, S. Guo, R. Zhang, P. Li, Y. Zhan, and W. Chen, "Cycle: Sustainable Off-Chain Payment Channel Network with Asynchronous Rebalancing," in *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2022, pp. 41–53. [Online]. Available: https://ieeexplore.ieee.org/document/9833795

Appendix

List of Acronyms

Table 8. KEY ACRONYMS AND DEFINITIONS

Acronym	Definition
A2A	Agent-to-Agent Protocol
AAMAS	Autonomous Agents and Multi-Agent Systems
ACL	Agent Communication Language
ACM	Association for Computing Machinery
AI	Artificial Intelligence
AMT	Amazon Mechanical Turk
ANP	Agent Network Protocol
ANS	Agent Naming Service
API	Application Programming Interface
AWS	Amazon Web Services
BDI	Belief-Desire-Intention
BPM	Business Process Management
CEUR-WS	CEUR Workshop Proceedings
CIKM	Conference on Information and Knowledge Management
DAI	Distributed Artificial Intelligence
DAO	Decentralized Autonomous Organization
DECP	Decentralized Ethical Consensus Protocol
DEXA	Database and Expert Systems Applications
DF	Directory Facilitator
DID	Decentralized Identifier
DOM	Document Object Model
EDOC	Enterprise Distributed Object Computing
ESWS	European Semantic Web Symposium
ETHOS	Ethical Technology and Holistic Oversight System
EU	European Union
FIPA	Foundation for Intelligent Physical Agents
FIRE	Flexible Integrated Reputation and Trust
FOCET	Functional Ontology of Context for Evaluating Trust
GDELT	Global Database of Events, Language, and Tone
GDPR	General Data Protection Regulation
GPT	Generative Pre-trained Transformer
HATP	Hierarchical Agent-based Task Planner
HTML	Hypertext Markup Language
HTN	Hierarchical Task Network
HTTP	Hypertext Transfer Protocol
I/O	Input/Output
IBM	International Business Machines
IBN	Intent-Based Networking
ICLR	International Conference on Learning Representations
ICMAS	International Conference on Multi-Agent Systems
IDE	Integrated Development Environment
JADE	Java Agent DEvelopment Framework
JSON	JavaScript Object Notation
JSON-LD	JavaScript Object Notation for Linked Data
JSON-RPC	JavaScript Object Notation Remote Procedure Call
KQML	Knowledge Query and Manipulation Language
LLM	Large Language Model
LNCS	Lecture Notes in Computer Science
LOKA	Layered Orchestration for Knowledgeable Agents
MAS	Multi-Agent Systems
MCP	Model Context Protocol
MDP	Markov Decision Process
MIT	Massachusetts Institute of Technology
NIST	National Institute of Standards and Technology

Acronym	Definition
OWASP	Open Web Application Security Project
OWL	Web Ontology Language
OWL-S	OWL for Services
P2P	Peer-to-Peer
PAAM	Practical Application of Intelligent Agents and Multi-Agent Technology
PDDL	Planning Domain Definition Language
PMLR	Proceedings of Machine Learning Research
PNSQC	Pacific Northwest Software Quality Conference
POMDP	Partially Observable Markov Decision Process
RAG	Retrieval-Augmented Generation
RDF	Resource Description Framework
RDFS	RDF Schema
REST	Representational State Transfer
RL	Reinforcement Learning
RPC	Remote Procedure Call
SDK	Software Development Kit
SHIQ	Description Logic with features S, H, I, and Q
SOAP	Simple Object Access Protocol
SOP	Standard Operating Procedure
SPARQL	SPARQL Protocol and RDF Query Language
SSE	Server-Sent Events
SSI	Self-Sovereign Identity
SW	Semantic Web
SWWS	Semantic Web and Web Services
TLS	Transport Layer Security
TOS	Terms of Service
ToT	Tree of Thoughts
UAIL	Universal Agent Identity Layer
UDDI	Universal Description, Discovery, and Integration
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VC	Verifiable Credential
W3C	World Wide Web Consortium
WI	Web Intelligence
WI-IAT	Web Intelligence and Intelligent Agent Technology
WIC	Web Intelligence Consortium
WIJ	Web Intelligence Journal
WoA	Web of Agents
WSDL	Web Services Description Language
WSIG	Web Service Integration Gateway
WSMO	Web Service Modeling Ontology
WWW	World Wide Web
XPath	XML Path Language
ZKP	Zero-Knowledge Proof