

Towards verifiable and trustworthy AI agents combining LLMs and classical agency

Silvia Di Giulio^{1,2}, Luigi Troiano^{1,2}, Massimo Bertolotti^{1,3} and Genoveffa Tortora¹

¹University of Salerno, Fisciano, Italy

²Kebula srl, Pellezzano, Italy

³SKY Italia, Milan, Italy

Abstract

The advent of Large Language Models (LLMs) has enabled a new class of agentic systems, significantly enhancing the flexibility, adaptability, and interactive capabilities of autonomous software agents operating in open and unstructured environments. However, this shift also introduces important challenges, as the inherently stochastic nature of generative models complicates verification, reproducibility, and robustness, ultimately limiting the ability to ensure consistent and reliable behaviour.

In this context, this paper investigates the integration of classical agent architectures and modern LLM-based systems as complementary paradigms for the design of intelligent agents. While LLM-driven approaches provide clear advantages in expressiveness and adaptability, their underlying mechanisms differ fundamentally from those of traditional formal frameworks. To characterise this divergence, we identify and analyse three key gaps between the two paradigms: (i) semantic and epistemological gap, concerning how knowledge and meaning are represented; (ii) architectural and control gap, addressing how behaviour is structured and guided; and (iii) verification and reliability gap, focusing on guarantees of correctness and robustness.

Building on this analysis, the paper reviews current hybrid approaches, including the integration of cognitive architectures with LLMs, world model augmentation, and neuro-symbolic methods, highlighting how these directions attempt to mitigate such limitations. Finally, we propose a multi-layered conceptual architecture that integrates LLMs with BDI (Belief-Desire-Intention) abstractions and explicit world models. By grounding neural outputs within an intermediate symbolic layer, the proposed framework combines the generative power of LLMs with the inspectability and controllability of classical systems, paving the way toward more reliable and trustworthy autonomous agents.

Keywords

Agentic AI, AI agents, LLM, Artificial Intelligence

1. Introduction

The recent emergence of agentic systems leveraging Large Language Models (LLMs) has significantly expanded the capabilities of autonomous software agents, enabling unprecedented levels of flexibility, adaptability, and natural interaction. By relying on LLMs, these agents can operate in open and unstructured environments, interpret complex inputs, and dynamically generate actions without the need for rigid predefined rules across a wide range of application domains [1]. However, this flexibility comes at a cost: the inherently non-deterministic nature of generative models introduces challenges in terms of verifiability, reproducibility, and debugging, making it difficult to guarantee consistent and reliable behaviour in real-world applications, especially in safety-critical settings [1, 2].

Current agentic platforms, including those widely adopted in industrial contexts, largely rely on techniques such as Chain-of-Thought (CoT) prompting [3], in which reasoning is guided through structured natural-language interactions. While effective in improving performance on complex tasks, these approaches remain fundamentally prompt-dependent and stochastic, due to the nature of LLMs, with behaviour that may vary significantly depending on input phrasing and context [4]. As a result,

WOA 2026: the 27th Workshop From Objects to Agents, June 15–17, 2026, Salerno, Italy

✉ sdi Giulio@unisa.it (S. Di Giulio); ltroiano@unisa.it (L. Troiano); mbertolotti@unisa.it (M. Bertolotti); tortora@unisa.it (G. Tortora)

🆔 0009-0004-7127-3441 (S. Di Giulio); 0000-0002-8304-8813 (L. Troiano); 0009-0004-6499-7126 (M. Bertolotti); 0000-0003-4765-8371 (G. Tortora)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

control over the internal reasoning process remains limited, potentially causing misalignment with user intent, and the systematic validation of agent decisions is often challenging [2].

In contrast, classical agent paradigms were built upon explicitly defined structures and formal semantics, where concepts such as beliefs, goals, and plans were represented through well-defined abstractions [5, 6, 7, 8, 9]. These systems offered strong guarantees in terms of transparency, inspectability, and verifiability, enabling developers to reason formally about agent behaviour. However, they were often characterized by rigidity, requiring substantial initial modelling effort and struggling to scale and adapt in dynamic and information-rich environments [1, 10].

This contrast reveals fundamental gaps between traditional and LLM-based approaches at the semantic, architectural, and verification levels. While classical agents rely on formal languages and explicit representations, LLM-based agents operate primarily through natural language and implicit, statistically induced structures.

This paper contributes along four main directions. First, it provides a structured analysis of the main gaps between classical and LLM-based agents, focusing on semantic, architectural, and verification aspects. Second, it reviews recent hybrid approaches through the lens of these gaps, identifying how different integration strategies attempt to reconcile symbolic and subsymbolic paradigms. Third, it proposes a conceptual multi-layer architecture for agentic systems that combines LLM-based reasoning with explicit symbolic representations, verification mechanisms, and world models. Fourth, it identifies network digital twins for CDN (Content Delivery Network), edge, and virtualized client/server infrastructures as a concrete and scientifically relevant domain in which this architecture can be instantiated and evaluated, thus turning the digital twin into an explicit world model for safe what-if analysis, causal diagnosis, and verifiable action planning. In this perspective, the future of agentic AI is not seen as the replacement of classical methods, nor vice versa, but rather as an evolution toward hybrid systems capable of combining formal structure with adaptive intelligence, ultimately enabling the development of agents that are both flexible and controllable.

The remainder of this paper is organized as follows. Section 2 reviews the main theoretical foundations of classical and LLM-based agents, highlighting their respective assumptions and design principles. Section 3 discusses the fundamental gaps between these two paradigms, with particular attention to semantic, architectural, and verification issues. Section 4 examines recent research directions aimed at bridging these gaps, including the integration of cognitive architectures and LLMs, the use of explicit world models, and symbolic approaches for controllable agency. Section 5 presents our conceptual multi-layer architecture, combining LLM-based interpretation, symbolic planning, external verification, and world-model-based simulation. Section 6 discusses network digital twins for CDN, edge, and virtualized client/server infrastructures as a concrete domain in which the proposed architecture can be instantiated and studied. Finally, Section 7 concludes the paper and outlines directions for future work.

2. Background: agentic theories and architectures

Although LLM-based agents have recently brought the notion of agency back to the centre of AI research, the concept of an intelligent agent is by no means new [11]. Its roots can be traced back to early work in logical AI, where representation, inference, and action were already conceived as tightly integrated, as exemplified by McCarthy’s *Programs with Common Sense* [12]. By the mid-1990s, the agent had become a well-established abstraction in artificial intelligence and computer science, with agent theory, agent architectures, and agent languages. This conceptual shift was notably formalised by Wooldridge and Jennings [9], who established a rigorous framework for the design and analysis of intelligent systems.

From this perspective, the recent success of LLM-based agents should not be interpreted as the emergence of agency in AI, but rather as a new technological instantiation within a long-standing research tradition. What distinguishes contemporary approaches is the integration of large-scale learned models, natural-language interfaces, and probabilistic mechanisms for reasoning and control.

2.1. Classical AI agents

Traditional methods for designing intelligent agents are primarily rooted in the symbolic and deliberative paradigm, characterised by explicit representations of knowledge and formally defined decision mechanisms. One category comprises reactive and behaviour based approaches, such as the subsumption architecture proposed by Brooks [5] who illustrated a reactive architecture in which simple modules directly map perceptions to actions without complex symbolic representations, favouring rapid and robust responses to environmental stimuli.

Alongside these, logical and rule-based approaches rely on declarative knowledge and explicit symbolic inferences. A key theoretical contribution was provided by Shoham [13], who introduced Agent Oriented Programming, modelling agents as entities with mental states (beliefs, goals, capabilities) described in a formal language. This enabled verifiable reasoning and a clear separation between knowledge and control.

A central role in the literature on agents is occupied by BDI (Belief-Desire-Intention) architectures, which combine reactivity and deliberation through explicit mental states. The original philosophical ideas on intentions and planning were formalised by Bratman [14], and subsequently developed in the field of agent theory by Rao and Georgeff [6, 7], providing abstract interpreters and executable architectures for rational agents based on beliefs, desires and intentions. An example of a theoretic formalization of a BDI agent programming language is AgentSpeak(L) [15], adopted in implementations such as Jason [16].

Finally, organization oriented and hybrid approaches extend the individual agent model to include social and coordination structures, supporting roles, norms, and missions within complex multi-agent systems. This shift from purely individual reasoning to normative MAS (Multi-Agent Systems) introduces explicit social and collective constraints that help regulate agent behaviour and promote coordination in decentralized environments.

Overall, these classical approaches share the use of formal structures and explicit reasoning mechanisms, ensuring properties such as verifiability, controllability and modularity, but they show limitations in terms of scalability and adaptability in open and dynamic contexts, especially when knowledge is not fully specified a priori.

2.2. LLM-based agents

In recent years, the emergence of agent methods based on Large Language Models (LLM) has shifted the field toward a predominantly subsymbolic (i.e. natural language) and partially emergent paradigm, where behaviour arises from the interaction between the model's latent knowledge and the provided context rather than from explicitly encoded rules. Foundational to this approach are structured prompting techniques, such as zero-shot and few-shot prompting [17], which demonstrate that task specification through natural language is possible without retraining the model, simply by providing the LLM with a few (or even zero) examples. To improve performance on complex tasks, researchers have introduced reasoning-elicitation strategies such as "scratchpads" for intermediate computation [18] and Chain of Thought (CoT) prompting [3], which encourage the model to articulate intermediate reasoning steps. These approaches have been further extended by frameworks like Tree of Thoughts (ToT) [19], enabling the exploration and evaluation of multiple candidate reasoning paths.

The operational capacity of LLM-based agents is improved through iterative loop architectures, such as the ReAct framework [20], which interleave reasoning and action execution. This acting capability is increasingly formalised through structured tool use, where models invoke external APIs, calculators, or search engines, often via schema-based interfaces. As highlighted by Mialon et al. [21] in their survey on Augmented Language Models (ALMs), such integration allows agents to overcome the limitations of purely parametric knowledge. A seminal contribution in this direction is Toolformer [22], which demonstrates how models can be trained to autonomously decide when and how to use external tools and incorporate their outputs into the generation process.

To support these operations, modern LLM-based agents incorporate memory mechanisms and in-

creasingly sophisticated orchestration layers. While short-term memory is bounded by the context window, long-term capabilities are typically enabled through Retrieval Augmented Generation (RAG) [23], often implemented via vector based semantic retrieval. Notably, works such as [24] show how combining LLMs with structured memory streams and retrieval mechanisms can give rise to persistent and context-aware agent behaviour, including planning and social interaction in simulated environments.

These components are integrated within orchestration frameworks such as LangChain [25] and Semantic Kernel [26], or more generally within modular control loops that manage state, planning, and tool execution. In such systems, the agent is no longer a monolithic entity but a compositional architecture in which prompting, reasoning, acting, and memory interact to produce coherent and adaptive behaviour.

Despite these advances, important challenges remain in terms of reliability, controllability, and systematic evaluation, including issues such as hallucinations, sensitivity to prompting variations, limited interpretability of internal processes, and the difficulty of ensuring robust and reproducible behaviour [1, 2, 4, 27].

3. The fundamental gaps between LLM agents and classical agency

The evolution from classical agents to those based on Large Language Models marks a profound transformation in how agency, reasoning, and planning are conceived, represented, and controlled. In classical systems, the agent is modelled through explicit and formalised abstractions (e.g. beliefs, desires, intentions, operational rules, roles, norms, and environmental artefacts) which constitute the semantic and architectural core of behaviour [6, 7, 9, 14]. These well-established frameworks provide declarative formalisms that support design, analysis, and formal verification, ensuring high levels of controllability, transparency, and reliability, even in complex multi-agent environments. Table 1 summarizes the main differences between classical agents and LLM-based agents across key dimensions, including representation, semantics, control, planning, interaction, verification, and programmability.

Table 1

Comparison between classical agents and LLM-based agents

Dimension	Classical agents	LLM-based agents
Representation	symbolic, explicit, formally structured	natural language based, distributed, context dependent
Semantics	compositional, logic grounded, interpretable	statistical, implicit, pattern based
Control	rule based or architecturally specified	prompt driven, loop based, externally orchestrated
Planning	algorithmic, model driven, explicit	emergent, context dependent, partially implicit
Interaction	messages, roles, norms, commitments	prompt based coordination, tool invocation, external protocols
Verification	formally defined and often provable	empirical, approximate, requiring external validation
Programmability	localized in explicit models or languages	distributed across prompts, tools, memory, and workflows

In contrast, LLM-based agents rely on a fundamentally different paradigm, in which behaviour is not fully specified a priori but arises from the interaction between the model’s learned representations and a surrounding ecosystem of prompts, control loops, tools, and memory mechanisms. This shift has fuelled renewed interest in notions such as ‘Agentic AI’ and ‘Multi-agent AI’, often used to describe autonomous generative systems. However, such terminology risks conflating recent technological advances with new theoretical foundations, potentially overlooking decades of research in classical agent theory [11]. As a result, the transition to LLM-based agents is not merely an extension of existing

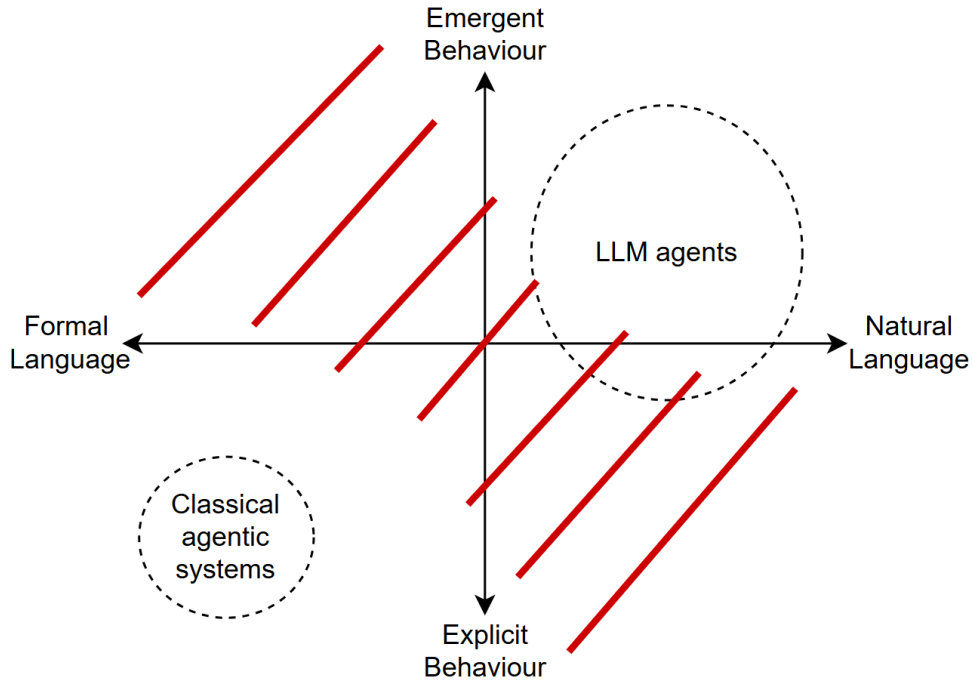


Figure 1: The current research gap and new conceptual space represented (in red) along two main axes: nature of language representation and behavioural specification.

paradigms, but involves a reconfiguration of how agents are specified, guided, and controlled in settings where reasoning is probabilistic and context-dependent rather than strictly logical.

This transformation introduces a set of fundamental gaps between the two paradigms, which can be analysed along three main dimensions: (i) semantic and epistemological, concerning how knowledge and meaning are represented; (ii) architectural and control-related, addressing how behaviour is structured and guided; and (iii) verification and reliability, focusing on guarantees of correctness and robustness.

These differences are summarised in Figure 1, which positions classical and LLM-based agents along two orthogonal dimensions: the nature of representation (from formal to natural language) and the locus of behavioural specification (from explicit to emergent). Classical systems occupy the region characterised by formal representations and explicitly defined behaviour, whereas LLM-based agents lie toward the opposite end, where behaviour is only partially specified and arises through interaction with linguistic context and external components.

The intermediate regions highlight the current lack of well-defined paradigms that systematically integrate these perspectives. This space represents a key research frontier, where the goal is to combine the interpretability and formal guarantees of symbolic approaches with the flexibility and expressiveness of LLM-based systems. The following subsections analyse these gaps in detail.

3.1. Semantic and epistemological gap

In classical systems, the semantics of agents is compositional, well-defined, and interpretable through explicit rules: beliefs, goals, intentions, commitments, and plans are formal entities grounded in precise logical frameworks. This enables not only the prediction of behaviour, but also reasoning about decisions and the application of formal verification techniques. Symbolic representations thus ensure transparency and inspectability, as each deliberative step can be traced and explained.

In LLM-based agents, by contrast, these notions often appear only as prompt-level constructs or descriptive labels, without corresponding to explicitly defined internal representations. The model’s internal state is implicitly distributed across its parameters and the input context, and meaning is derived from statistical associations learned during training, rather than from explicitly grounded symbols. From this perspective, model outputs reflect probabilistic patterns in data rather than explicitly modelled

knowledge or reasoning processes [4]. Consequently, semantics is data-driven and only partially controllable, and reasoning is better understood as an induced behaviour rather than a formally specified process.

This shift reframes the epistemological perspective: whereas classical systems focus on how an agent should reason, LLM-based agents raise the question of how reasoning can be elicited and controlled within a pre-trained model.

3.2. Architectural, control and planning gap

In classical systems, the organisation of state, reasoning, planning, and communication is explicitly defined within the architecture. Planning relies on algorithms operating over symbolic representations of the environment and objectives, while coordination is governed by formally specified rules and organisational models [5, 6, 7, 8, 9]. This structural clarity supports predictability and facilitates systematic analysis of agent behaviour.

In LLM-based agents, these elements are distributed across the model and its operational environment. Behaviour emerges from the interaction between prompts, orchestration layers, external tools, memory components, and retrieval mechanisms. Planning and decision-making are not always the result of explicit algorithms, but can take the form of context-dependent reasoning traces expressed in natural language [3, 20]. While such traces may be partially observable, their variability and lack of formal grounding limit their robustness. Consequently, control is achieved indirectly, through prompt design and external coordination mechanisms rather than fully specified internal processes: the central challenge remains the principled integration of these intelligent generative components into coherent and reliable architectures.

3.3. Verification and reliability gap

The final dimension concerns guarantees of correctness and reliability. In classical systems, correctness can be formally assessed: rules can be verified against conditions, plans can be checked for consistency, and execution can be validated against safety constraints. This enables the development of deterministic and trustworthy agents suitable for critical applications.

In LLM-based agents, correctness is typically evaluated empirically. Outputs may vary even under similar conditions due to stochastic generation and sensitivity to input formulation of LLMs [1, 4]. Ensuring reliability therefore requires additional layers of monitoring, validation, and structured orchestration. The absence of fully explicit semantics and deterministic behaviour complicates both transparency and accountability, particularly in high-stakes scenarios.

4. Some Recent Research Directions

The growing adoption of LLM-based agents has highlighted a fundamental tension between expressive power and controllability. While recent approaches demonstrate remarkable flexibility and adaptability, their reliance on implicit representations, probabilistic reasoning, and emergent behaviour raises concerns regarding transparency, predictability, and formal guarantees. As discussed in the previous sections, these limitations are not merely engineering challenges, but reflect deeper mismatches between subsymbolic generation and the requirements of structured agency.

From this perspective, the problem is not only how to improve performance, but how to define forms of agency that remain interpretable and governable. Recent work such as [28] frames this challenge in terms of abstraction, arguing that effective agent design requires intermediate layers that allow humans to understand, design, and control behaviour despite the complexity of the underlying models.

In parallel to these conceptual considerations, a number of recent research directions have begun to explore how to reintroduce structure, grounding, and verification into LLM-based systems. Rather than replacing existing paradigms, these approaches aim to complement and constrain generative models through architectural, representational, or procedural mechanisms. The common objective is to move

from loosely orchestrated agent pipelines toward more principled designs, in which behaviour can be analysed, guided, and, at least partially, validated.

Some of these efforts focus on architectural decomposition and modularity. For instance, [29] emphasises the role of explicit world models and the separation between perception, reasoning, and action, as a way to ground behaviour in structured and inspectable representations. Other works instead explore what can already be achieved within largely LLM-centric frameworks. Approaches such as [2, 24] demonstrate that complex behaviours—such as planning, memory usage, and social interaction—can emerge from the interaction between language models and relatively simple orchestration mechanisms. At the same time, these works highlight important limitations, as their underlying processes remain largely heuristic and lack formal semantics or verifiable guarantees.

These limitations have contributed to a growing emphasis on reliability and evaluation in LLM-based systems. In this context, techniques such as tool use, structured prompting, and externalised reasoning (e.g., program synthesis or retrieval-based methods) can be an initial step toward more controllable and inspectable forms of behaviour, even though they are often developed in a task-specific manner.

Taken together, these developments point toward a research landscape that is both active and still fragmented. Rather than converging on a single solution, current work explores multiple, partially complementary directions. Among these, three representative lines of research can be identified: (i) the integration of LLMs within cognitive architectures to provide explicit control structures; (ii) the use of grounded and externalised world models; and (iii) symbolic approaches that combine probabilistic reasoning with explicit representations and verification mechanisms. While not exhaustive, these directions delineate an emerging design space in which the flexibility of LLMs is progressively balanced with the need for structure, interpretability, and reliability.

4.1. Integrating agent architectures like BDI and LLMs

A particularly interesting direction concerns the incorporation of Large Language Models within established agent architectures, including BDI.

The explicit representation of beliefs, goals, and intentions of the BDI model, together with its deliberative control loop, provides a natural scaffold for integrating data-driven components while preserving a degree of interpretability and control. As highlighted in recent exploratory studies on the integration of LLMs with cognitive and agent architectures [30], LLMs can be viewed as complementary modules within a larger reasoning system, rather than as standalone agents.

Within the BDI deliberative cycle, LLMs (and more broadly, AI techniques) can be integrated at multiple stages [31]. In the sensing phase, LLMs could act as semantic interpreters of unstructured or ambiguous inputs, supporting the update of beliefs from textual or noisy data. In the deliberation and planning phases, LLMs can assist in generating or refining plans based on current beliefs and goals, leveraging their contextual reasoning capabilities. In the execution phase, they can support action selection, parameterisation, or interaction in environments requiring linguistic flexibility. Additionally, LLMs can be employed as meta-reasoning components, for example to evaluate alternative plans, provide explanations, or assist in resolving conflicts between intentions.

Existing approaches in the literature such as [32, 33], demonstrate the feasibility of combining LLMs with BDI reasoning. Nevertheless, these solutions typically employ LLMs in a limited or task-specific manner, such as supporting plan generation or natural language interpretation. Up to our knowledge, there is a lack of architectures that integrate LLMs in a principled and structured way across the entire BDI cycle, while preserving the core properties of BDI systems, including explicit semantics, deliberative control, and verifiability.

A further limitation concerns current programming models for BDI agents. Traditional frameworks such as Jason, 2APL, and Jadex provide explicit constructs for beliefs, goals/desires, intentions, rule-based reasoning, and plan libraries. These features support transparency, verifiability, and fine-grained control. However, such frameworks are generally better suited to relatively structured environments, where knowledge can be represented symbolically and agent behaviour can be engineered explicitly.

Integrating LLMs effectively requires more flexible representations and execution models: beliefs may

need to incorporate unstructured or probabilistic content and planning should support dynamic generation and adaptation at runtime. Current BDI languages offer limited support for these requirements, as they lack native abstractions for interacting with LLMs as part of the reasoning process.

From the perspective of the three gaps identified earlier, this line of research provides a broad but still partial response. Regarding the semantic and epistemological gap, BDI architectures preserve explicit constructs for beliefs, goals, and intentions, thereby providing a structured context in which LLM outputs can be interpreted rather than treated as self-sufficient reasoning processes. Regarding the architectural, control and planning gap, the BDI deliberative cycle offers an explicit control structure in which LLMs can be assigned circumscribed roles, such as interpretation, plan generation, or meta-reasoning, instead of driving the entire agent through prompts alone. Regarding the verification and reliability gap, the explicit representation of goals, plans, and constraints can still support monitoring and partial validation, even when some reasoning steps are delegated to probabilistic models.

At the same time, these benefits remain limited in current approaches: dynamically generated plans must still be checked against the agent’s goals and constraints, and LLM outputs may require filtering or correction. As a result, the guarantees provided by classical BDI systems are difficult to preserve when reasoning is partially delegated to probabilistic components.

Overall, integrating LLMs within BDI and other practical reasoning frameworks is a promising direction, but existing solutions remain fragmented and largely ad hoc. Principled architectures and programming languages for a seamless integration of LLMs across the full agent lifecycle are still missing, especially if one aims to preserve the foundational properties of classical BDI systems.

4.2. World models for agentic systems

A second research direction focuses on the introduction of explicit world representations as a core component of LLM-based agents.

The underlying idea is to make the representation of the environment explicit and inspectable, rather than implicitly encoded in model parameters and context. This aligns with the view that effective agency requires a structured representation of the world to support reasoning, planning, and interaction. In this perspective, world models are treated as distinct components that mediate between perception, reasoning, and action [29]. This separation is closely related to the grounding problem: while LLMs exhibit strong formal and linguistic capabilities, they lack a stable connection between internal representations and external entities [27].

Different approaches instantiate this idea in different ways. Some works focus on external world representations, such as Digital Twins, which provide high-fidelity models of real environments and enable simulated execution before acting [34]. Others explore lighter-weight or simulated environments, where persistent memory structures support coherent long-term behaviour, as in [24]. A complementary line of work addresses the internal limitations of LLM reasoning, proposing to augment it with explicit planning mechanisms. For instance, [35] employs the LLM both as a reasoning agent and as a world model, combined with Monte Carlo Tree Search [36] to explore alternative reasoning trajectories.

From the perspective of the identified gaps, these approaches primarily address the semantic, architectural, and verification dimensions of the problem, by introducing explicit representations of state that improve inspectability, by structuring the relation between perception, reasoning, and action through an explicit world model, and by enabling the evaluation of candidate actions and future outcomes before execution.

Nevertheless, current solutions are heterogeneous and differ significantly in their assumptions and scope. External world models raise challenges related to grounding and synchronisation with real environments, while internal approaches still rely on approximations of state transitions. As a result, a unified and general framework for integrating world models within LLM-based agents remains an open issue.

A particularly relevant instantiation of this perspective emerges in the context of network digital twins. In such settings, the world model is not merely an internal abstraction, but an explicit and continuously calibrated representation of the operational environment, supporting simulation, counterfactual

reasoning, and pre-deployment validation. This makes network digital twins a natural domain for studying verifiable agentic systems, as they require the integration of observability, causal diagnosis, symbolic planning, and safety-aware execution under realistic operational constraints.

4.3. Agents with symbolic representation

A complementary research direction focuses on the reintroduction of symbolic representations within LLM-based agents, with the goal of improving structure, controllability, and verifiability [37]. While often framed under the umbrella of neuro-symbolic AI, this line of work more broadly includes approaches that integrate explicit symbolic abstractions (such as programs, formal languages, or constraint systems) into otherwise subsymbolic architectures.

The core idea underlying these approaches is to complement subsymbolic generation with intermediate representations that make parts of the reasoning process explicit and inspectable. Rather than relying solely on free-form text generation, these methods introduce structured artefacts that mediate between natural language and action.

A first class of approaches externalises parts of the reasoning process into executable symbolic representations. For instance, Program-Aided Language Models (PAL) [38] translate natural language problems into programs, delegating execution to an external interpreter. This separation allows the LLM to focus on problem decomposition, while the symbolic runtime ensures correctness in computation, reducing errors that typically arise in purely textual reasoning.

A related line of work explores the use of formal languages to constrain and supervise LLM-based planning. Approaches such as [39] introduce explicit task specifications (automata) that guide the generation process and ensure that produced plans satisfy predefined structural properties. These methods improve predictability by restricting the space of admissible behaviours.

Other techniques constrain the generation process itself. Methods such as constrained decoding and specialized query or programming languages for LLMs (e.g. [40] and [41]) enforce grammatical and logical constraints during autoregressive inference. Rather than filtering outputs after generation, these approaches restrict the set of admissible tokens at each step, ensuring compliance with a predefined schema while preserving the model’s expressive capabilities.

From the perspective of the gaps identified earlier, symbolic approaches contribute primarily to addressing the semantic and verification challenges of LLM-based agents, by reintroducing explicit structure and enabling partial forms of validation. However, current solutions remain heterogeneous and often tailored to specific representations or tasks, and do not yet provide general-purpose frameworks in which symbolic and neural components are seamlessly integrated across the full agent lifecycle.

4.4. Conclusions and Open Challenges

Building on the research directions discussed in the previous sections, Table 2 provides a synthesis of the current landscape. It summarises how the identified approaches mitigate the core gaps (Section 3) while also highlighting the critical challenges that remain open. As the table illustrates, although each direction offers significant advancements, none provides a complete solution on its own. Motivated by these unresolved challenges, the following section theorises a comprehensive agentic architecture designed to synergistically integrate Large Language Models, cognitive architectures, symbolic representations, and explicit world models into a unified framework.

5. A Conceptual Architecture for a new class of AI Agents

The limitations of both purely symbolic agents and purely LLM-driven systems suggest the need for a more principled class of architectures that should not be understood as simple compositions of prompts, tools, and symbolic modules, but as structured cognitive systems in which heterogeneous components play distinct yet tightly coordinated roles. A suitable reference model for this class of systems can be formulated by combining the BDI paradigm with recent proposals from cognitive

Table 2

Summary of recent research directions, how they address the gaps identified in Section 3, and open challenges.

Research Direction	Mainly addressed gaps	Open challenges
Integration of agent architectures (e.g., BDI) and LLMs	<ul style="list-style-type: none"> - Semantic gap - Architectural gap - Verification gap 	<ul style="list-style-type: none"> - Achieving full integration of LLMs across all stages of the BDI cycle. - Developing programming languages that natively blend BDI constructs and prompting. - Reliable verification mechanisms for dynamically generated plans.
World models for agentic systems	<ul style="list-style-type: none"> - Semantic gap - Architectural gap - Verification gap 	<ul style="list-style-type: none"> - Establishing a principled and scalable integration between the agent’s core controller and the explicit world model. - Ensuring consistent grounding over time.
Agents with symbolic representations	<ul style="list-style-type: none"> - Semantic gap - Verification gap 	<ul style="list-style-type: none"> - Integration of symbolic and neural components across the full agent lifecycle.

architectures and autonomous intelligence research. On the one hand, BDI provides an explicit account of deliberation through the representation of beliefs, desires, intentions, and plans. On the other hand, recent architectural perspectives emphasise the importance of functionally separating executive control, memory, action generation, predictive world modelling, and evaluation. Taken together, these views indicate that a hybrid agent should be organised as a multi-layered architecture in which symbolic and sub-symbolic processes are coordinated through explicit interfaces.

The core operational loop of this architecture is detailed in Figure 2. Following the input, a *neural semantic layer*, made of an LLM (that could also use CoT [3], for example), processes the unstructured input to perform intent interpretation and goal selection. The *Knowledge Base* (KB) of the architecture maintains explicit representations of beliefs, goals, and intentions: it acts as the central repository for symbolic artifacts, including plans, actions, tools, and a memory. The memory component can be made of a long term memory using a RAG [23] and an episodic memory to store eventual failures.

Rather than delegating execution implicitly to the LLM, the architecture implements a dual-mode response mechanism:

- **Exploitation (proven plans):** The system first queries the Knowledge Base. If a verifiable plan for the specified intent already exists, it is immediately routed for symbolic plan execution in the real world. A subsequent positive response from the environment triggers a final KB update, closing the safe execution loop.
- **Exploration (novel plans):** If a suitable plan does not exist, a neuro-symbolic loop is activated. A secondary LLM, specializing in plan generation, translates the abstract goal into a candidate plan. These intermediate symbolic artifacts render the reasoning transparent and amenable to rigorous validation.

Before deployment in the real world, candidate plans pass through a *Verification layer*. This layer is functionally differentiated into two modules: a symbolic verification module, which checks consistency and constraint satisfaction, and a world model simulation module, which utilizes a Digital Twin or a domain model for grounded prediction, counterfactual evaluation, and look-ahead evaluation. If this predictive evaluation identifies a constraint violation, it feeds a verification error signal back to the LLM plan generator for iterative repair, mitigating the risk of executing hallucinated or physically impossible plans. Successful plans are simultaneously routed for execution and saved to the Knowledge Base, thus accumulating reliable knowledge over time.

This organization renders uncertainty manageable through explicit loops for failure. Following real-world execution, the system evaluates the outcome. If the response from the environment is

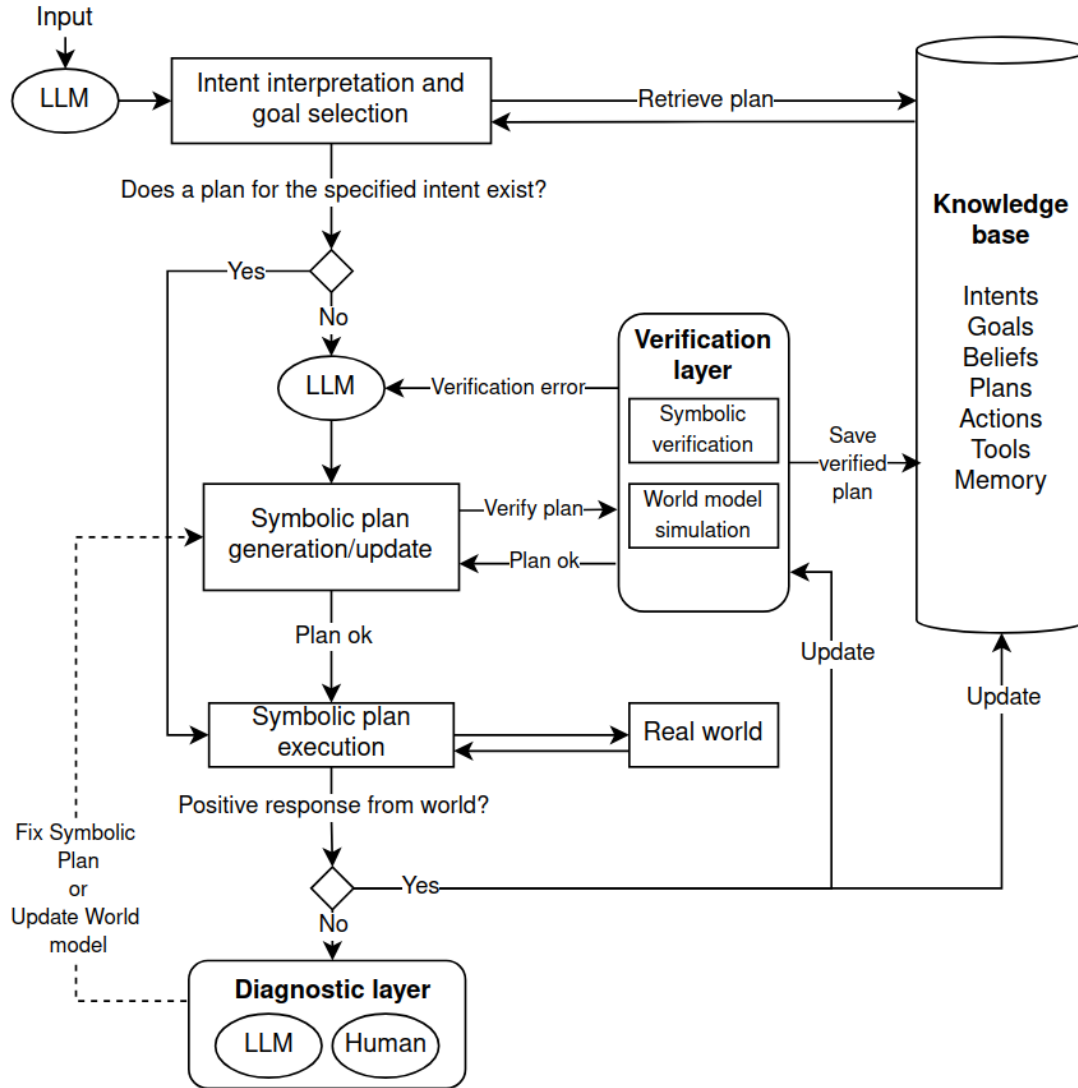


Figure 2: A conceptual agent architecture combining BDI-driven deliberation, neural inference (LLM), externalized verification and simulation loops, and explicit diagnostic failure loops.

negative, the process transitions to a dedicated *Diagnostic layer*. This layer, supported by an LLM or *human in the loop*, analyzes the failure data to either update the generated symbolic plan or to update the world model.

Overall, the architecture establishes a controlled neuro-symbolic loop: the LLM proposes, the symbolic layer formalizes, the world model simulates, and the diagnostic layer aims to ensure resilience. By explicitly decoupling heuristic generation from logical and physical validation, this design seeks to move AI agents away from purely probabilistic approximations towards more grounded deliberation. Within this framework, the language model is not treated as a standalone oracle, but is instead integrated as a bounded cognitive subsystem subject to external verification. While this structural discipline inevitably introduces additional computational and architectural overhead, it provides a necessary foundation for mitigating hallucination and execution risks in dynamic and critical environments. Ultimately, this approach does not eliminate the inherent uncertainties of open-world deployment, but it ensures that the agent's behavior and knowledge base can be systematically audited and refined, tying autonomous actions to explicit constraints rather than latent statistical associations alone.

6. Network Digital Twins as a Concrete Domain for Verifiable Agency

The theoretical gaps and architectural requirements identified in the previous sections find a particularly relevant application domain in the management of modern network infrastructures, including CDN, edge, cloud, and virtualized client/server systems. In these environments, service quality does not depend on a single component, but emerges from the interaction of multiple highly interdependent elements, such as distributed caches, routing policies, peering strategies, load balancing mechanisms, application-level logic, client behaviour, and local network conditions. Recent literature on network digital twins has highlighted how these environments require explicit models that support monitoring, experimentation, and closed-loop control under realistic operational constraints [42, 43].

In such settings, a digital twin can play the role of an explicit world model for agentic systems. Rather than relying solely on implicit representations encoded in prompts or model parameters, the agent can ground its deliberation in a structured and inspectable model of topology, resources, workloads, policies, and failure modes. This perspective is particularly relevant in networked environments, where the effects of an action may be non-local, delayed, and difficult to predict. By integrating simulation into the agentic loop, the twin makes it possible to perform what-if analysis, compare alternative interventions, and validate candidate actions before their execution in the real system [43, 44].

This is especially important in CDN and virtualized client/server infrastructures, where performance depends not only on network-side mechanisms, but also on realistic client behaviour. In practice, user-perceived quality is shaped by the interaction among cache hit and miss dynamics, adaptive bitrate selection, retry logic, session patterns, device heterogeneity, and geographical distribution. For this reason, a useful digital twin should not treat clients and servers as abstract traffic sources, but should explicitly model their behaviour as first-class entities. Existing work on CDN simulation and adaptive video streaming shows the importance of combining infrastructure-level models with realistic application- and client-side dynamics [44, 45].

From the perspective of the architecture proposed in Section 5, the digital twin provides the operational grounding required to transform LLM-based reasoning into verifiable action. The LLM acts as a semantic interface for interpreting unstructured intents, operator requests, incident reports, and troubleshooting contexts. However, its outputs are not executed directly. Instead, they are translated into symbolic plans, policy candidates, or operational constraints, which can be checked for consistency and then evaluated through simulation against the current state of the infrastructure. In this way, the digital twin bridges the semantic gap by grounding interpretation in explicit state, the architectural gap by supporting structured reasoning over future outcomes, and the verification gap by making candidate actions testable before deployment.

A further requirement in this domain concerns diagnosis. In complex infrastructures, telemetry signals such as latency spikes, QoE (Quality of Experience) degradation, or increased error rates may be correlated without revealing the actual operational cause. Therefore, root-cause analysis should not be based only on heuristic association, but should instead aim to produce evidence-based explanations grounded in causal reasoning. This is particularly important in agentic systems, where decisions should be justified and traceable rather than merely plausible. Recent work on causal inference for online service systems provides useful methodological foundations for this perspective [46].

Similarly, action selection in these environments must remain subject to explicit safety constraints. Candidate changes may have a broad blast radius, interact with time-sensitive service commitments, or produce unintended degradations in other parts of the system. For this reason, any agentic platform operating in closed loop should incorporate mechanisms for safety-aware validation, including rollback conditions, change windows, operational limits, and SLO/SLA (Service Level Objective, Service Level Agreement) preservation. In this respect, broader work on safe decision-making under constraints offers a useful conceptual basis for designing agentic systems that remain effective without sacrificing reliability [47].

These considerations suggest that network digital twins are not merely an application example of the architecture proposed in this paper, but a particularly meaningful testbed for studying verifiable agency in realistic operational conditions. They provide an explicit world model, a rich source of observability

Table 3

Mapping between the conceptual hybrid architecture and an agentic platform for network digital twins.

Architectural component	Conceptual role	Instantiation in the network digital twin platform
Neural semantic layer	Interpretation of unstructured inputs and intent selection	LLM-based processing of operator requests, incidents, runbooks, and high-level goals
Knowledge base	Explicit storage of beliefs, goals, plans, tools, and memory	Topology, configurations, policies, change history, runbooks, telemetry-derived features, validated plans
Symbolic plan layer	Formalization of candidate actions into inspectable representations	Routing policies, cache rules, scaling plans, remediation workflows, operational constraints
Verification layer	Logical and constraint-based validation of plans	Safety checks on SLO/SLA, blast radius, rollback conditions, and change windows
World model simulation	Grounded prediction and evaluation of candidate plans	Digital twin of nodes, links, workloads, client behaviour, failures, and application logic
Diagnostic layer	Analysis of failure and iterative correction	Root-cause analysis, causal evidence extraction, plan revision, and twin recalibration

data, and a natural setting in which symbolic planning, causal diagnosis, simulation-based validation, and safety-aware execution must be tightly integrated. For this reason, they offer a concrete domain in which hybrid agentic architectures can be instantiated, analysed, and progressively validated, while preserving the balance between neural flexibility and formal control that motivates this work.

This domain also highlights a number of open challenges, including the trade-off between twin fidelity and operational cost, the modelling of realistic client and server behaviour, the integration of causal diagnosis, and the design of safety-aware closed-loop control mechanisms. These challenges confirm that network digital twins are not merely an application example, but a meaningful setting in which the requirements of verifiable agency become particularly evident.

A further implication of this perspective is that trustworthy agency should not be reduced to a single generative component. In the network digital twin setting, a more principled solution is to decompose the overall system into specialized agents coordinated through a shared knowledge base and a common world model. This organizational choice is consistent with the broader claim of this paper: reliable agency emerges not from unconstrained generation alone, but from the structured interaction of components with distinct responsibilities and explicit interfaces.

In practical terms, such a decomposition may include agents devoted to telemetry analysis, causal diagnosis, policy optimization, capacity planning, client-behaviour modelling, simulation orchestration, and safety enforcement. Rather than acting as isolated modules, these components contribute to a common deliberative and verification loop: some interpret signals and anomalies, others formulate and compare candidate actions, while others assess their operational admissibility through simulation and constraint checking. Such decomposition improves modularity, inspectability, and controllability, while preserving the flexibility of LLM-based interpretation where it is most useful. Table 3 summarizes how the main components of the conceptual hybrid architecture can be instantiated in an agentic platform for network digital twins, highlighting the correspondence between abstract functions and operational modules.

7. Conclusion

In this work, we have argued that the current evolution of agentic AI should not be understood as a replacement of classical agency by LLM-based approaches, but rather as a transition toward hybrid

systems capable of combining formal structure, explicit control, and adaptive intelligence. Starting from this perspective, we have analyzed the main semantic, architectural, and verification gaps between classical agents and contemporary LLM-based agents, showing how these differences affect the way agency is represented, implemented, and evaluated.

On this basis, we have reviewed a set of emerging research directions aimed at bridging these gaps, including the integration of LLMs within cognitive architectures, the use of explicit world models, and the reintroduction of symbolic representations for more controllable forms of reasoning and action. These directions suggest that the future of agentic systems lies not in purely prompt-driven orchestration, nor in a return to fully rigid symbolic approaches, but in the design of architectures where neural and symbolic components play complementary and explicitly coordinated roles.

The conceptual architecture proposed in this paper should be understood as a step in this direction. By combining LLM-based interpretation with explicit symbolic plans, external verification mechanisms, and world-model-based simulation, the framework aims to make agent behaviour more inspectable, controllable, and reliable, while preserving the flexibility that has made LLM-based systems attractive in open and unstructured environments.

The discussion of network digital twins as a telecommunications use case has further illustrated how these requirements become particularly evident in realistic operational domains. In such contexts, the need for explicit grounding, simulation-based validation, causal diagnosis, and safety-aware execution shows why trustworthy agency cannot be reduced to unconstrained generative reasoning alone. More generally, this example suggests that hybrid agentic architectures are especially relevant in settings where autonomous decisions must remain explainable, auditable, and bounded by operational constraints.

Several challenges remain open. Future work should further investigate how to formalize the interaction between symbolic and neural components, how to validate dynamically generated plans in a principled way, and how to integrate world models that remain both computationally tractable and sufficiently faithful to the real environment. Additional work is also needed on programming abstractions, evaluation methodologies, and domain-specific instantiations that can turn conceptual architectures of this kind into robust and reusable agentic systems.

Ultimately, the contribution of this work is not to provide a definitive solution, but to outline a conceptual space in which verifiable and trustworthy agency may be pursued beyond the opposition between classical AI and LLM-based methods. In this sense, the development of future agentic systems may depend less on choosing between symbolic and generative paradigms, and more on understanding how to combine them within architectures that are both flexible and governable.

Declaration on Generative AI

During the preparation of this work, ChatGPT and Gemini have been used in order to: Improve writing style. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

- [1] S. Deshmukh, M. Kuri, V. Vanarote, M. Nimbalkar, P. Chandre, S. Yadav, Agentic ai: A survey of autonomous agents, architectures, and emerging applications, in: 2025 IEEE 5th International Conference on ICT in Business Industry & Government (ICTBIG), IEEE, 2025, pp. 1–8.
- [2] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, M. Zhang, J. Wang, S. Jin, E. Zhou, et al., The rise and potential of large language model based agents: A survey, *Science China Information Sciences* 68 (2025) 121101.
- [3] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, et al., Chain-of-thought prompting elicits reasoning in large language models, *Advances in neural information processing systems* 35 (2022) 24824–24837.
- [4] E. M. Bender, T. Gebru, A. McMillan-Major, S. Shmitchell, On the dangers of stochastic parrots: Can language models be too big?, in: *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, 2021, pp. 610–623.
- [5] R. Brooks, A robust layered control system for a mobile robot, *IEEE journal on robotics and automation* 2 (2003) 14–23.
- [6] M. Georgeff, A. Rao, Modeling rational agents within a bdi-architecture, in: *Proc. 2nd Int. Conf. on Knowledge Representation and Reasoning (KR'91)*. Morgan Kaufmann, of, 1991, pp. 473–484.
- [7] A. S. Rao, M. P. Georgeff, et al., Bdi agents: from theory to practice., in: *Icmas*, volume 95, 1995, pp. 312–319.
- [8] F. Dignum, Autonomous agents with norms, *Artificial intelligence and law* 7 (1999) 69–79.

- [9] M. Wooldridge, N. R. Jennings, Intelligent agents: Theory and practice, *The knowledge engineering review* 10 (1995) 115–152.
- [10] M. Georgeff, B. Pell, M. Pollack, M. Tambe, M. Wooldridge, The belief-desire-intention model of agency, in: *International workshop on agent theories, architectures, and languages*, Springer, 1998, pp. 1–10.
- [11] V. Botti, Agentic ai and multiagentic: Are we reinventing the wheel?, *arXiv preprint arXiv:2506.01463* (2025).
- [12] J. McCarthy, Programs with Common Sense, Technical Report, 1960.
- [13] Y. Shoham, Agent-oriented programming, *Artificial intelligence* 60 (1993) 51–92.
- [14] M. Bratman, Intention, plans, and practical reason (1987).
- [15] A. S. Rao, Agentspeak (I): Bdi agents speak out in a logical computable language, in: *European workshop on modelling autonomous agents in a multi-agent world*, Springer, 1996, pp. 42–55.
- [16] R. H. Bordini, J. F. Hübner, M. J. Wooldridge, M. J. Wooldridge, Programming multi-agent systems in AgentSpeak using Jason, volume 8, Wiley Online Library, 2007.
- [17] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, *Advances in neural information processing systems* 33 (2020) 1877–1901.
- [18] M. Nye, A. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, C. Sutton, A. Odena, Show your work: Scratchpads for intermediate computation with language models, *ArXiv abs/2112.00114* (2021).
- [19] S. Yao, D. Yu, J. Zhao, I. Shafran, T. Griffiths, Y. Cao, K. Narasimhan, Tree of thoughts: Deliberate problem solving with large language models, *Advances in neural information processing systems* 36 (2023) 11809–11822.
- [20] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. R. Narasimhan, Y. Cao, React: Synergizing reasoning and acting in language models, in: *The eleventh international conference on learning representations*, 2022.
- [21] G. Mialon, R. Dessi, M. Lomeli, C. Nalmpantis, R. Pasunuru, R. Raileanu, B. Rozière, T. Schick, J. Dwivedi-Yu, A. Celikyilmaz, et al., Augmented language models: a survey, *arXiv preprint arXiv:2302.07842* (2023).
- [22] T. Schick, J. Dwivedi-Yu, R. Dessi, R. Raileanu, M. Lomeli, E. Hambro, L. Zettlemoyer, N. Cancedda, T. Scialom, Toolformer: Language models can teach themselves to use tools, *Advances in neural information processing systems* 36 (2023) 68539–68551.
- [23] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, et al., Retrieval-augmented generation for knowledge-intensive nlp tasks, *Advances in neural information processing systems* 33 (2020) 9459–9474.
- [24] J. S. Park, J. O’Brien, C. J. Cai, M. R. Morris, P. Liang, M. S. Bernstein, Generative agents: Interactive simulacra of human behavior, in: *Proceedings of the 36th annual acm symposium on user interface software and technology*, 2023, pp. 1–22.
- [25] H. Chase, Langchain, 2022. URL: <https://github.com/langchain-ai/langchain>.
- [26] Microsoft, Semantic kernel, 2023. URL: <https://github.com/microsoft/semantic-kernel>.
- [27] E. Pavlick, Symbols and grounding in large language models, *Philosophical Transactions of the Royal Society A* 381 (2023) 20220041.
- [28] A. Ricci, S. Mariani, F. Zambonelli, S. Burattini, C. Castelfranchi, The cognitive hourglass: Agent abstractions in the large models era, in: *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems, AAMAS ’24, International Foundation for Autonomous Agents and Multiagent Systems*, 2024, p. 2706–2711.
- [29] Y. LeCun, et al., A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27, *Open Review* 62 (2022) 1–62.
- [30] O. J. Romero, J. Zimmerman, A. Steinfeld, A. Tomasic, Synergistic integration of large language models and cognitive architectures for robust ai: An exploratory analysis, in: *Proceedings of the AAAI Symposium Series*, volume 2, 2023, pp. 396–405.
- [31] R. H. Bordini, A. El Fallah Seghrouchni, K. Hindriks, B. Logan, A. Ricci, Agent programming in the cognitive era, *Autonomous Agents and Multi-Agent Systems* 34 (2020) 37.
- [32] M. Jang, Y. Yoon, J. Choi, H. Ong, J. Kim, A structured prompting based on belief-desire-intention model for proactive and explainable task planning, in: *Proceedings of the 11th International Conference on Human-Agent Interaction, HAI ’23, Association for Computing Machinery*, New York, NY, USA, 2023, p. 375–377.
- [33] L. Frering, G. Steinbauer-Wagner, A. Holzinger, Integrating belief-desire-intention agents with large language models for reliable human–robot interaction and explainable artificial intelligence, *Engineering Applications of Artificial Intelligence* 141 (2025) 109771.
- [34] A. Hasan, D. T. Nguyen, Integrating agentic ai and digital twins for intelligent decision-making systems, *Array* 29 (2026) 100721.
- [35] S. Hao, Y. Gu, H. Ma, J. Hong, Z. Wang, D. Wang, Z. Hu, Reasoning with language model is planning with world model, in: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 2023, pp. 8154–8173.
- [36] L. Kocsis, C. Szepesvári, Bandit based monte-carlo planning, in: J. Fürnkranz, T. Scheffer, M. Spiliopoulou (Eds.), *Machine Learning: ECML 2006*, Springer Berlin Heidelberg, 2006, pp. 282–293.
- [37] S. B. Hakim, M. Adil, A. Velasquez, H. H. Song, Neuro-symbolic agentic ai: Architectures, integration patterns, applications, open challenges and future research directions, *Computer Science Review* 60 (2026) 100902.
- [38] L. Gao, A. Madaan, S. Zhou, U. Alon, P. Liu, Y. Yang, J. Callan, G. Neubig, Pal: program-aided language models, in: *Proceedings of the 40th International Conference on Machine Learning, ICML’23, JMLR.org*, 2023.
- [39] Z. Li, W. Hua, H. Wang, H. Zhu, Y. Zhang, Formal-llm: Integrating formal language and natural language for controllable llm-based agents, *arXiv preprint arXiv:2402.00798* (2024).
- [40] L. Beurer-Kellner, M. Fischer, M. Vechev, Prompting is programming: A query language for large language models, *Proc.*

ACM Program. Lang. 7 (2023).

- [41] B. T. Willard, R. Louf, Efficient guided generation for large language models, arXiv preprint arXiv:2307.09702 (2023).
- [42] A. Hakiri, A. Gokhale, S. B. Yahia, N. Mellouli, A comprehensive survey on digital twin for future networks and emerging internet of things industry, *Computer Networks* 244 (2024) 110350.
- [43] G. Nardini, G. Stea, Using network simulators as digital twins of 5g/b5g mobile networks, in: *Proceedings of the 2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2022, pp. 584–589.
- [44] K. Stamos, G. Pallis, A. Vakali, D. Katsaros, A. Sidiropoulos, Y. Manolopoulos, Cdnsim: A simulation tool for content distribution networks, *ACM Transactions on Modeling and Computer Simulation* 20 (2010) 1–40.
- [45] H. Mao, R. Netravali, M. Alizadeh, Neural adaptive video streaming with pensieve, in: *Proceedings of ACM SIGCOMM '17*, 2017, pp. 197–210.
- [46] M. Li, Z. Li, K. Yin, X. Nie, W. Zhang, K. Sui, D. Pei, Causal inference-based root cause analysis for online service systems with intervention recognition, in: *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22)*, 2022, pp. 3230–3240.
- [47] J. García, F. Fernández, A comprehensive survey on safe reinforcement learning, *Journal of Machine Learning Research* 16 (2015) 1437–1480.