AITHOS

Next Generation of Autonomous Agents for Distributed Systems and Digital Assets

AITHOS Labs Team
team@aithos.xyz
January 2025

Keywords:
Distributed Systems, AI Agents, Agent Platform, Blockchain, Digital Assets, Decentralized Architecture, Scalability, Security

ABSTRACT

The unprecedented rise of AI agents marks a pivotal moment in the evolution of software, promising a future of autonomous systems capable of reshaping industries and redefining human-computer interaction. However, current agents are hampered by their limited capabilities, and inability to operate effectively within distributed, trustless environments in order to solve more meaningful tasks. This whitepaper presents a novel vision for the next generation of AI agents in order to unlock their full potential. We propose a blueprint for decentralized, secure, and collaborative agents that will not only overcome the limitations of today's systems but also enable a new era of intelligent, autonomous entities capable of navigating the complex landscapes of digital assets and beyond.

TABLE OF CONTENTS

1. INTRODUCTION

The rise of Artificial Intelligence (AI) agents marks a paradigm shift in software development, promising a future where autonomous entities can independently perform complex tasks, reshape industries, and revolutionize human-computer interaction. These agents, driven by sophisticated algorithms and machine learning models, are increasingly capable of automating processes, providing personalized services, and engaging in dynamic decision-making. However, despite their advancements, current AI agents are fundamentally limited by their inherent architecture. The vast majority operate within centralized systems, subject to single points of control and failure. Furthermore, their capabilities are often siloed, lacking the interoperability and collaborative potential needed to tackle complex, real-world problems that span multiple domains and require coordination with other agents.

The increasing need for autonomous agents operating in decentralized, trustless environments is becoming ever more apparent. Consider the management of digital assets, for example: the emerging landscape of blockchain, decentralized finance (DeFi), and tokenized assets demands agents that can operate securely and transparently without reliance on central authorities. Existing AI agents often fall short of fulfilling these requirements due to their inherent constraints in security, transparency, and autonomy within these complex, decentralized settings. The current lack of robust, decentralized, and collaborative AI agents is a crucial bottleneck in realizing the full potential of both distributed systems and the rapidly evolving digital asset space. This paper aims to address these limitations, presenting a novel vision for the next generation of AI agents.

2. FOUNDATIONS

2.1 Blockchain and DeFi

Since the launch of Bitcoin in 2009 [1] and Ethereum in 2015 [2], the blockchain ecosystem has undergone a transformative evolution. What began as a decentralized system for peer-to-peer transactions has grown into a complex industry encompassing diverse technologies, infrastructures, and applications. Today, we see a proliferation of blockchains, including Layer 1 networks (e.g., Ethereum, Solana, Sui, etc), Layer 2 scaling solutions (e.g., Optimistic Rollups, zkRollups) designed to address specific use cases.

Blockchain technology's fundamental attributes offer significant advantages. Open systems provide universal access and eliminate intermediaries, while composability enables developers to stack and integrate protocols to create novel applications. Transparency ensures accountability by making all transactions publicly verifiable, and self-custody empowers users to retain full control over their assets, eliminating reliance on centralized entities.

The financial applications of blockchain, collectively referred to as Decentralized Finance (DeFi), span a wide array of products, including faster and cheaper payments, decentralized borrowing and lending platforms, automated market makers for trading, derivatives such as futures and options, and new innovations like tokenized real-world assets and Decentralized Physical Infrastructure Networks (DePIN). These innovations have unlocked global financial participation, democratizing access to products and services traditionally limited to centralized institutions.

Growing problems

The rapid growth of this ecosystem has introduced new challenges. The landscape has become increasingly fragmented, with users navigating a bewildering array of wallets, protocols, and chains. Keeping up with the ever-evolving DeFi ecosystem requires specialized knowledge and constant attention, making it inaccessible to many. Security concerns also loom large, with vulnerabilities in smart contracts, phishing attacks, and scams leading to billions in losses. The high risk of user error, such as sending funds to the wrong address, adds another layer of complexity. This underscores the need for solutions that can simplify and streamline interaction with this transformative yet daunting technology.

2.2 The Rise of AI

Artificial intelligence has been a field of scientific exploration since the 1940s, evolving through decades of theoretical advances and practical breakthroughs. Early systems focused on symbolic reasoning and expert systems designed to mimic human decision-making in specific domains. The advent of deep learning marked a paradigm shift, enabling machines to learn complex patterns from vast datasets and unlocking applications in fields such as image recognition, natural language processing, and autonomous systems.

In 2017, Google introduced the transformer architecture [3], revolutionizing the way AI models process sequential data. Transformers formed the backbone of large language models (LLMs), culminating in innovations such as OpenAI’s ChatGPT, which achieved widespread public adoption in late 2022. This marked the beginning of a new AI revolution, characterized by a surge of interest, investment, and rapid advancements across the field.

Today, AI extends beyond text generation to encompass image recognition, generative art, code generation, and more. These advancements are converging to enable highly sophisticated use cases, such as end-to-end automation of tasks, enhanced data analysis, and real-time interaction with complex systems. The versatility and power of AI have positioned it as a transformative force across industries, laying the groundwork for its integration into decentralized systems like blockchain.

2.3 The Case for Agents and Blockchain together

The intersection of AI agents and blockchain represents a paradigm shift in how complex systems can be managed and optimized. By leveraging the strengths of both technologies, we can address challenges that exceed human capabilities while unlocking entirely new possibilities.

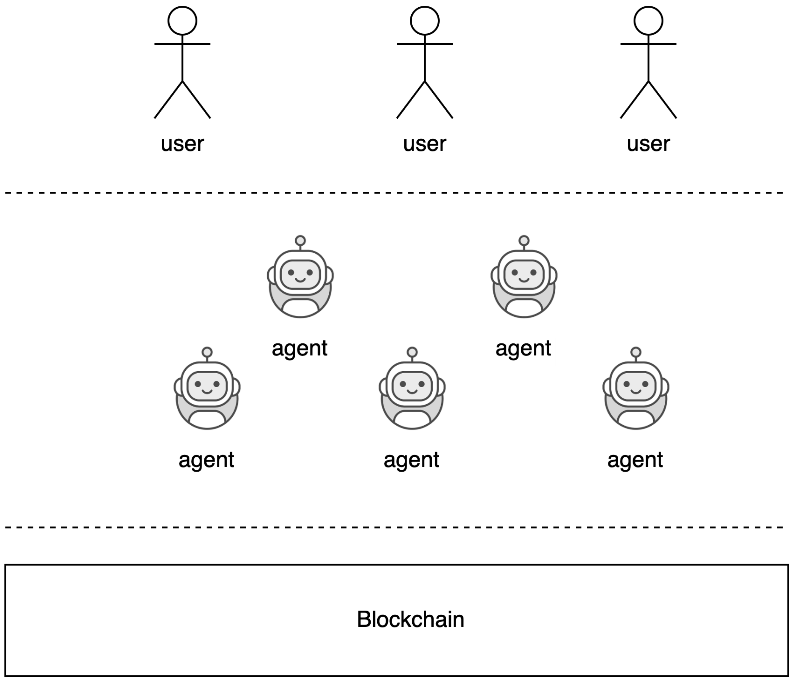
- **Handling Complexity Beyond Human Capacity**: The blockchain ecosystem’s complexity is often overwhelming for individual users. AI agents can act as intermediaries, analyzing options, automating decision-making, and mitigating risks in a space fraught with volatility and security challenges. They can also streamline interactions, ensuring users make informed choices without requiring deep technical expertise.
- **Faster, Cheaper, More Efficient Transactions**: AI agents, powered by blockchain, can facilitate transactions at unprecedented speed, enabling real-time payments, trading, and cross-border settlements. The programmability of smart contracts allows agents to execute predefined actions based on triggers, ensuring efficiency and accuracy. They can also plan execution strategies in order to optimize for cost, speed or other important metrics.
- **Micropayments and Agent Economies**: Blockchain’s ability to process micropayments opens the door to agent-to-agent economies. AI agents can exchange data, services, and resources autonomously, utilizing tokens or cryptocurrencies as a medium of value transfer. These transactions can form the basis of decentralized economic systems, where agents operate with their own tokens, incentives, and governance models.
- **The Importance of Open Data**: Blockchain's inherent transparency ensures that data is openly accessible, providing a fertile ground for AI agents to thrive. Access to on-chain data allows agents to derive insights, verify information, and make decisions in a decentralized manner. This symbiosis

enables trustless collaboration between agents and users, paving the way for robust decentralized applications.

Together, AI agents and blockchain have the potential to redefine industries, automate inefficiencies, and enable a future where intelligent systems interact seamlessly with decentralized technologies. These synergies will form the basis for the innovative platform proposed in this paper, empowering a new era of intelligent, decentralized ecosystems.

2.4. Application Scenarios

The integration of AI agents with blockchain technology creates a powerful synergy, enabling automation, optimization, and innovation across a wide spectrum of use cases. These agents can handle tasks of varying complexity, from simplifying user interactions to orchestrating intricate decentralized workflows. Below are several application scenarios that demonstrate the breadth and transformative potential of this integration:



1. Simplifying DeFi Interaction
DeFi protocols are notoriously complex and require significant knowledge and constant attention. AI agents can:

- Optimize Borrowing and Lending: Analyze market conditions to select the best platforms for borrowing or lending assets, dynamically adjusting strategies to maximize yield.
- Automate Yield Farming: Monitor liquidity pools and staking opportunities, reinvesting rewards or reallocating assets to more lucrative options in real-time.
- Facilitate Token Swaps: Identify the most favorable trading routes across decentralized exchanges (DEXs) while minimizing fees and slippage.

2. Wallet and Portfolio Management
Managing crypto wallets and diversified portfolios can be challenging for even experienced users. AI agents can:

- Monitor Wallets: Track balances, gas fees, and token performance, alerting users to potential risks such as low reserves for transaction fees.
- Rebalance Portfolios: Automatically adjust asset allocations based on market trends, risk tolerance, and predefined goals.
- Secure Wallets: Enhance wallet security by identifying potential phishing attempts, unauthorized access, or suspicious transactions.

3. Advanced Trading Strategies
AI agents can unlock sophisticated trading capabilities that were previously reserved for experts:

- Execute Algorithmic Trading: Implement high-frequency trading strategies, leveraging on-chain data and real-time market analysis.
- Arbitrage Opportunities: Identify price differences across exchanges and execute trades to capture profit in a decentralized manner.
- Risk Management: Automatically set stop-loss orders, hedge positions, or diversify exposure during market volatility.

4. Launching and Managing Tokens
For creators and entrepreneurs, AI agents can streamline the process of launching and managing tokens:

- Token Creation: Automate the technical aspects of creating ERC-20 or other blockchain-compliant tokens, complete with audits to ensure security.
- Marketing and Airdrops: Coordinate token distribution campaigns, targeting specific demographics or communities to maximize impact.
- Governance Participation: Engage in decentralized governance, casting votes or proposing changes based on predefined criteria or user preferences.

5. Automating Strategies and Monitoring Events
Blockchain’s transparency provides a wealth of actionable data that agents can monitor and act upon:

- Strategy Automation: Implement trading, staking, or lending strategies that execute based on market triggers such as price thresholds or interest rate changes.

- Event Monitoring: Track blockchain events like smart contract upgrades, token unlocks, or major governance proposals, ensuring timely responses.
- Liquidation Prevention: Monitor collateralized positions and take preemptive actions to avoid liquidations in lending protocols.

6. Crowdfunding and Project Launches
AI agents can democratize access to capital and streamline the development of decentralized projects:

- Crowdfunding Campaigns: Manage decentralized fundraising efforts, ensuring compliance with regulations and optimizing campaign visibility.
- DAO Formation: Automate the creation of Decentralized Autonomous Organizations (DAOs), including setting up governance rules, treasury management, and community onboarding.
- Project Orchestration: Coordinate between multiple agents to handle development, deployment, marketing, and community management for blockchain projects.

7. Collaborative Agent Ecosystems
AI agents can cooperate with other specialized agents, amplifying their collective capabilities:

- Multimodal Problem Solving: For example, one agent may analyze on-chain data, another could execute smart contracts, while a third handles natural language communication with users or other agents.
- Cross-Chain Operations: Enable seamless interactions between different blockchains, facilitating interoperability and liquidity aggregation.
- Complex Goal Achievement: Agents can collaborate to execute multi-step goals, such as analyzing market trends, launching a token, and managing liquidity provisioning across chains.

8. Risk Management and Compliance
Managing risk and adhering to regulations are critical in decentralized systems:

- Risk transparency: Integrating with decentralized identity solutions and reputation systems.
- Insurance Coverage: Monitor for potential exploits or smart contract vulnerabilities and purchase decentralized insurance coverage preemptively.
- Crisis Response: Execute predefined strategies during black swan events, such as rapid asset liquidation or debt consolidation.

9. Enabling Micropayment Economies
The scalability and programmability of blockchain enable new economic models driven by agents:

- Data Marketplaces: Agents can buy, sell, or share data in exchange for micropayments, creating decentralized data ecosystems.
- Service Marketplaces: Agents can offer services, such as running computations or validating data, autonomously in exchange for tokens.

10. Decentralized Intelligence Networks

Agents can aggregate and process decentralized data to offer insights and services:

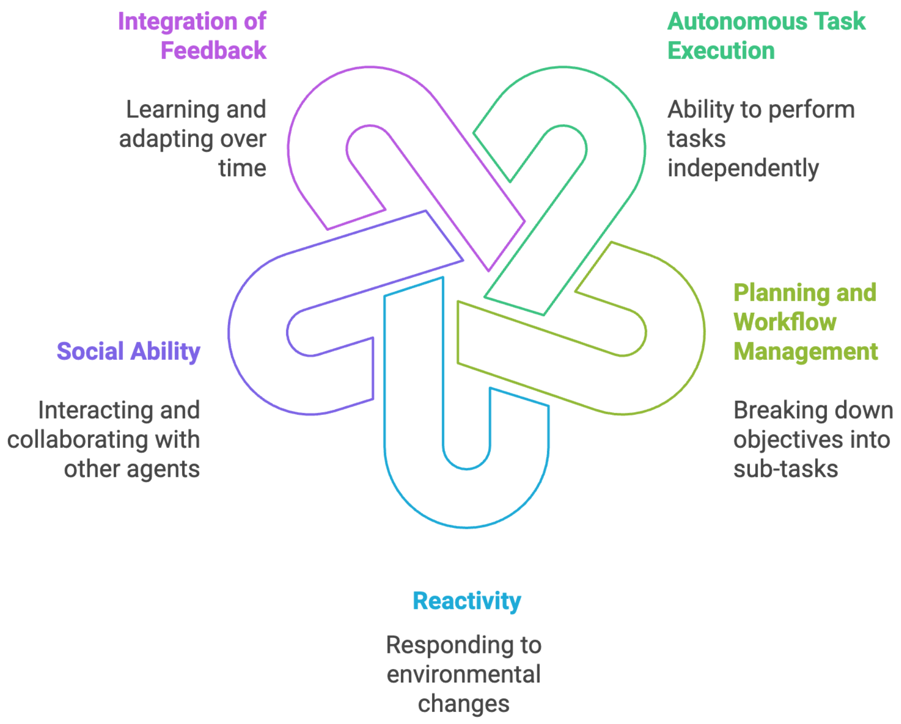
- Oracle Integration: Fetch and verify off-chain data for use in smart contracts.
- Predictive Analytics: Analyze historical blockchain data to forecast trends and guide user decisions.
- Crowdsourced Intelligence: Aggregate inputs from multiple agents to derive consensus-based recommendations.

By combining the automation and intelligence of AI agents with the transparency and trustlessness of blockchain, these scenarios showcase the potential to reshape how individuals and organizations interact with decentralized systems. This fusion not only simplifies access but also unlocks new economic paradigms and opportunities, paving the way for a more efficient, secure, and innovative decentralized future.

3. DEFINING AUTONOMOUS AGENTS

There are a lot of debates about what constitutes agents. Given the heated market, people are incentivized to call just about anything an agent. What used to be called a script or even the coffee machine at the office is now agentic. Anthropic has made a good attempt at differentiating agents from scripts and workflows [4] although it doesn't expand on how to differentiate agents with various levels of capabilities. We hereby propose a more standardized way to define what is an agent and to rate their capabilities based on a multi-level system.

3.1 Key Characteristics of Agents

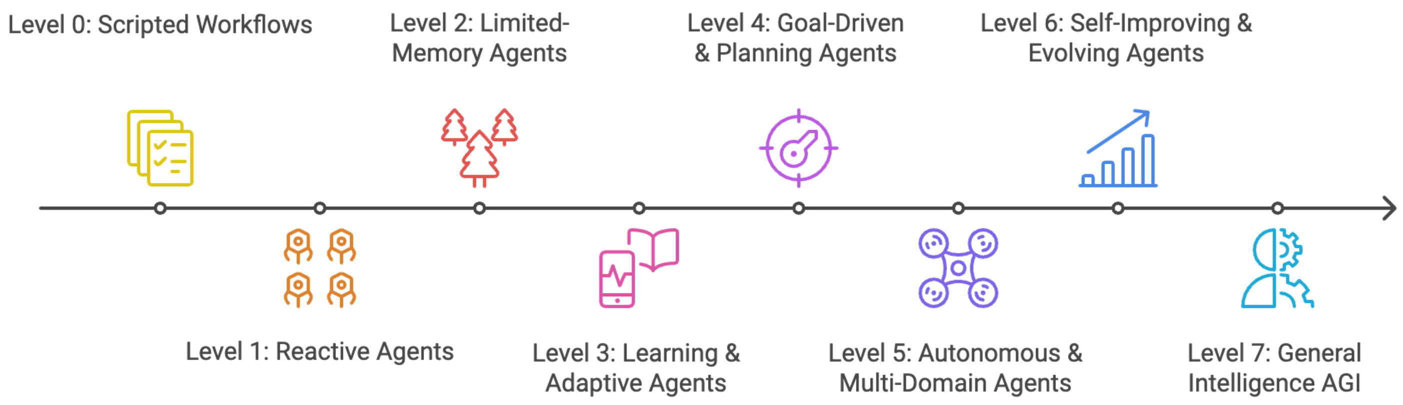

Integration of Feedback
Learning and adapting over time
Autonomous Task Execution
Ability to perform tasks independently
Planning and Workflow Management
Breaking down objectives into sub-tasks
Social Ability
Interacting and collaborating with other agents
Reactivity
Responding to environmental changes

- **Autonomous Task Execution**: The agent can independently perform a predefined or dynamically assigned task from initiation to completion without continuous human intervention, demonstrating the ability to independently handle the steps needed to reach a specific objective.
- **Planning and Workflow Management**: The agent is capable of breaking down complex objectives into sub-tasks, generating structured plans, and managing workflows to efficiently achieve the target goals, coordinating actions over time and sequencing activities in a logical order.
- **Reactivity (tools, environment)**: The agent can perceive, analyze, and respond to changes in its environment or use available tools in a timely and appropriate way, dynamically altering its actions based on the input it receives from its surroundings.

- **Social Ability (communication with other agents):** The agent is equipped to interact with other agents, exchange information, coordinate actions, and engage in collaborative problem-solving, facilitating the formation of agent networks and complex distributed systems.
- **Integration of Feedback (memory, adaptation):** The agent can learn from its experiences, integrate past data, and adapt its behavior over time through internal memory mechanisms and by incorporating feedback from both the environment and other agents, continuously improving its decision-making processes.

3.2 Classification of Agent Levels of Intelligence (CALI)

We propose the Classification of Agent Levels of Intelligence (CALI) as one way to rank agents according to how advanced they are. As we'll see in the following section, most agents deployed today fall in the lower end of this spectrum.


Level 0: Scripted Workflows
Level 1: Reactive Agents
Level 2: Limited-Memory Agents
Level 3: Learning & Adaptive Agents
Level 4: Goal-Driven & Planning Agents
Level 5: Autonomous & Multi-Domain Agents
Level 6: Self-Improving & Evolving Agents
Level 7: General Intelligence AGI

Level 0: Scripted Workflows

- **Description**: Non-adaptive workflows or automation scripts. They follow predetermined steps and lack any autonomy, intelligence, or learning.
- **Key Characteristics**:
  - Purely rule-based or sequential execution (e.g., bash scripts, RPA macros).
  - No memory or context-tracking; cannot deviate from the scripted path.
  - Not typically considered "agents" since there is no real decision-making.
- **Examples**: Basic pipeline automation, cron jobs, IFTTT workflows.

Level 1: Reactive Agents

- **Description**: Agents that respond directly to immediate inputs. They have **no internal memory** of past states and make decisions based solely on current sensory data or stimuli.
- **Key Characteristics**:
  - Stateless or very limited state; no longer-term context.
  - Simple perception-action mappings (e.g., if condition X, then do Y).
  - Unable to plan or learn over time; purely reactive.
- **Examples**:
  - Thermostats adjusting temperature in real-time.
  - Basic chatbots with purely keyword-based triggers.

Level 2: Limited-Memory Agents

- **Description**: Agents that can store and use **short-term memory** (e.g., recent interactions or states) to influence current actions. This introduces basic contextual awareness but not broad or long-term learning.
- **Key Characteristics**:
  - Maintains a buffer of recent events or observations.
  - Adapts its immediate responses based on limited history.
  - Still lacks robust long-term adaptation or generalization.
- **Examples**:
  - Basic game AIs that recall the last few moves of an opponent.
  - Chatbots that reference a short conversation history.

Level 3: Learning & Adaptive Agents

- **Description**: Agents capable of **improving over time** by learning from data or experience using machine learning techniques. Though they can generalize within a given domain, their abilities remain **bounded** by their initial design or training paradigm.
- **Key Characteristics**:
  - Employ supervised, unsupervised, or reinforcement learning to refine decision policies.
  - Retain knowledge over longer periods (beyond short memory buffers).
  - Continually improve performance within a **predefined domain**.
- **Examples**:
  - Spam filters that adapt to new email patterns.
  - Recommendation engines learning from user preferences.
  - Simple RL agents optimizing their reward functions in a single environment.

Level 4: Planning and Reasoning Agents

- **Description**: Agents that explicitly **formulate and pursue goals**, often using **planning** or **reasoning** techniques. They can break down tasks into sub-goals and use search or symbolic methods to achieve objectives.
- **Key Characteristics**:
  - Possess an internal representation of goals or objectives.
  - Leverage planning algorithms (e.g., search trees, symbolic reasoning) to select actions over a longer horizon.
  - Operate within a (often domain-specific) set of knowledge or constraints.
- **Examples**:
  - Scheduling agents that map out step-by-step tasks to complete a project.
  - Navigation systems that plan routes based on known maps and constraints.
  - Complex puzzle-solving AIs that strategically plan sequences of moves.

Level 5: Multi-Domain Agents

- **Description**: Agents capable of **independent operation** across a broader range of tasks or environments, often with minimal human oversight. They can adapt to novel situations within **related** or **partially known** domains.
- **Key Characteristics**:
  - Handle partially observable or dynamic environments with minimal explicit intervention.
  - Transfer or generalize knowledge across **multiple** related tasks.
  - Set sub-goals autonomously and manage resources or constraints over long time horizons.
- **Examples**:
  - Autonomous vehicles that navigate complex, real-world roads.
  - Advanced game-playing AIs (e.g., AlphaGo, AlphaZero) that adapt strategies for different but related games.
  - Virtual assistants that coordinate calendars, messages, and tasks with little prompting.

Level 6: Self-Improving & Evolving Agents

- **Description**: Agents that can **modify their own learning mechanisms** and discover **new capabilities** beyond their original design. This often involves meta-learning (learning how to learn) or architecture search.
- **Key Characteristics**:
  - Actively reconfigure internal models or algorithms to optimize performance.
  - Can spot knowledge gaps and autonomously acquire **entirely new** skills or data sources.
  - Exhibit emergent behaviors as a result of deep introspection and self-reflection on learning processes.
- **Examples**:
  - Systems that automatically design and train new neural network architectures.
  - AI that identifies limitations in its current approach and experiments with novel solutions.

- Agents capable of “self-play” to generate new strategies without explicit human guidance.

Level 7: General Intelligence / AGI

- **Description**: A **theoretical** level representing Artificial General Intelligence—comparable to or surpassing human intelligence across **all** domains. This may (or may not) include sentience, consciousness, or self-awareness, depending on one’s definitions.
- **Key Characteristics**:
  - Performs well on **any** intellectual task humans can do, potentially better.
  - Demonstrates broad adaptability, creative problem-solving, and abstract reasoning across disjoint fields.
  - May exhibit **self-awareness** or consciousness (highly debated and not yet realized).
- **Examples**:
  - Hypothetical AGI that could learn entirely new scientific disciplines, generate novel hypotheses, and independently push the frontiers of knowledge.
  - Sci-fi portrayals of “true” AI with human-like or superior cognition.

3.3 How the Levels Build on Each Other

1. **From Non-Adaptive to Reactive**: Level 0 to 1 introduces basic decision-making in response to the environment.
2. **From No Memory to Short-Term Memory**: Level 1 to 2 adds a memory buffer that enables minimal context-based actions.
3. **From Contextual to Fully Learning**: Level 2 to 3 expands an agent’s ability to **improve** over time using data/experience.
4. **From Learning to Planning**: Level 3 to 4 distinguishes agents that **formulate explicit goals** and multi-step plans rather than just learning reactive or policy-based rules.
5. **From Single-Domain to Multi-Domain Autonomy**: Level 4 to 5 broadens the scope, letting agents handle more complex or varied environments with minimal human guidance.
6. **From Autonomous to Self-Evolving**: Level 5 to 6 sees agents adapting their **own internal architectures** and discovering new capabilities independently.
7. **From Self-Evolving to AGI**: Level 6 to 7 represents the leap to **general** (and possibly conscious) intelligence, currently theoretical and not fully achieved.

In the next section we will examine the current states of agents, their limitations and how they rank based on our STAR system.

4. LIMITATIONS OF CURRENT AI AGENTS

The first generation of so-called agents in the crypto space were limited in both functionality and autonomy, often falling short of what could be considered true agents. These early examples included entities such as memecoins, virtual companions, Twitter personas, automated news reporters, and trading signal bots. While these applications gained some traction, they highlighted several significant shortcomings in the design and implementation of AI agents.

4.1 Key Limitations

1. **Centralized Control**
    - These agents were typically deployed on centralized cloud platforms like AWS, under the full control of their creators or hosting platforms.
    - Users had no assurance of the agent's autonomy, and there was often no guarantee that the agent would act in users' best interests.
2. **Shallow Integration with LLMs**
    - Many agents featured only basic integrations with large language models (LLMs), often limited to generating simple text outputs or responding to predefined prompts.
    - They lacked sophisticated reasoning, decision-making, or context-awareness capabilities.
3. **Limited Communication Channels**
    - User interaction was primarily restricted to platforms like Twitter (now X.com), Discord, or similar social media platforms.
    - These channels offered limited interactivity and failed to support complex workflows or task execution.
4. **Minimal or Nonexistent Memory Systems**
    - Most agents had no memory systems or only rudimentary short-term memory, preventing them from retaining context or learning from past interactions.
    - This lack of memory led to repetitive and shallow behavior, reducing their usefulness in long-term or evolving tasks.
5. **No Learning or Improvement**
    - These agents were static by design, unable to refine their strategies or improve over time based on user interactions or task outcomes.
    - As a result, they often became obsolete or ineffective in dynamic environments like the crypto and DeFi space.
6. **Transparency and Auditability Issues**
    - Most agents operated as black boxes, with no transparency in their operations or decision-making processes.
    - Users had no way to verify the integrity or correctness of an agent's actions, leading to potential risks of manipulation or misuse.
7. **Lack of Guardrails and Safeguards**
    - Early agents lacked robust mechanisms to prevent bugs, errors, or malicious behavior.

  - This absence of guardrails made them prone to unintended consequences, such as financial losses, data breaches, or exploitative practices.

According to the CALI system, which evaluates agents on a scale of developmental levels, most current agents are classified at Level 0 (static, rule-based systems), Level 1 (basic LLM integration), or Level 2 (capable of simple workflows but lacking autonomy).

Very few agents have progressed beyond these levels, falling short of true autonomy or adaptability.

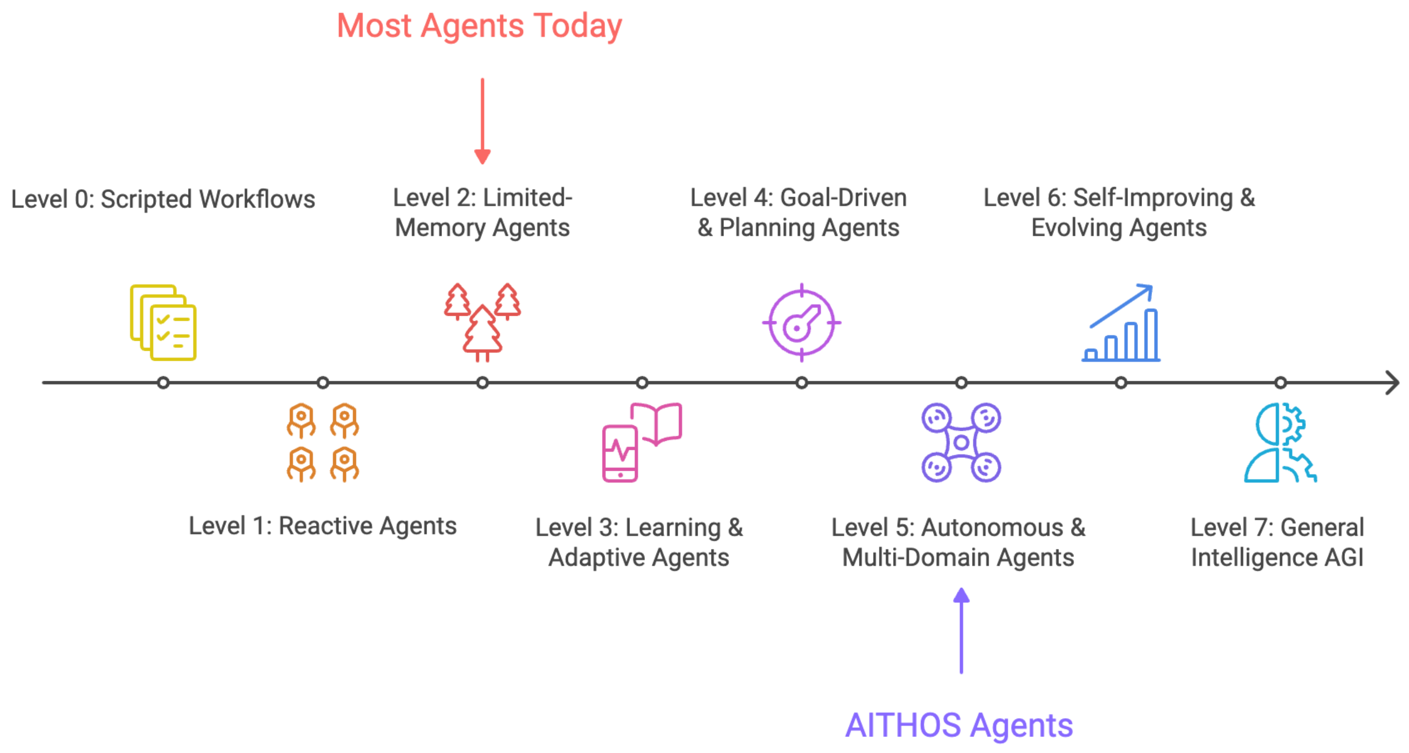
4.2 Broader Implications

The limitations of these early agents have significant implications:

- Trust Deficit: Users remain skeptical about delegating critical tasks to agents due to their lack of transparency and control.
- Missed Opportunities: Static and rudimentary agents are unable to unlock the full potential of automation in the rapidly evolving crypto and DeFi ecosystems.
- Risk Exposure: Without proper safeguards, these agents expose users and platforms to operational, financial, and reputational risks.

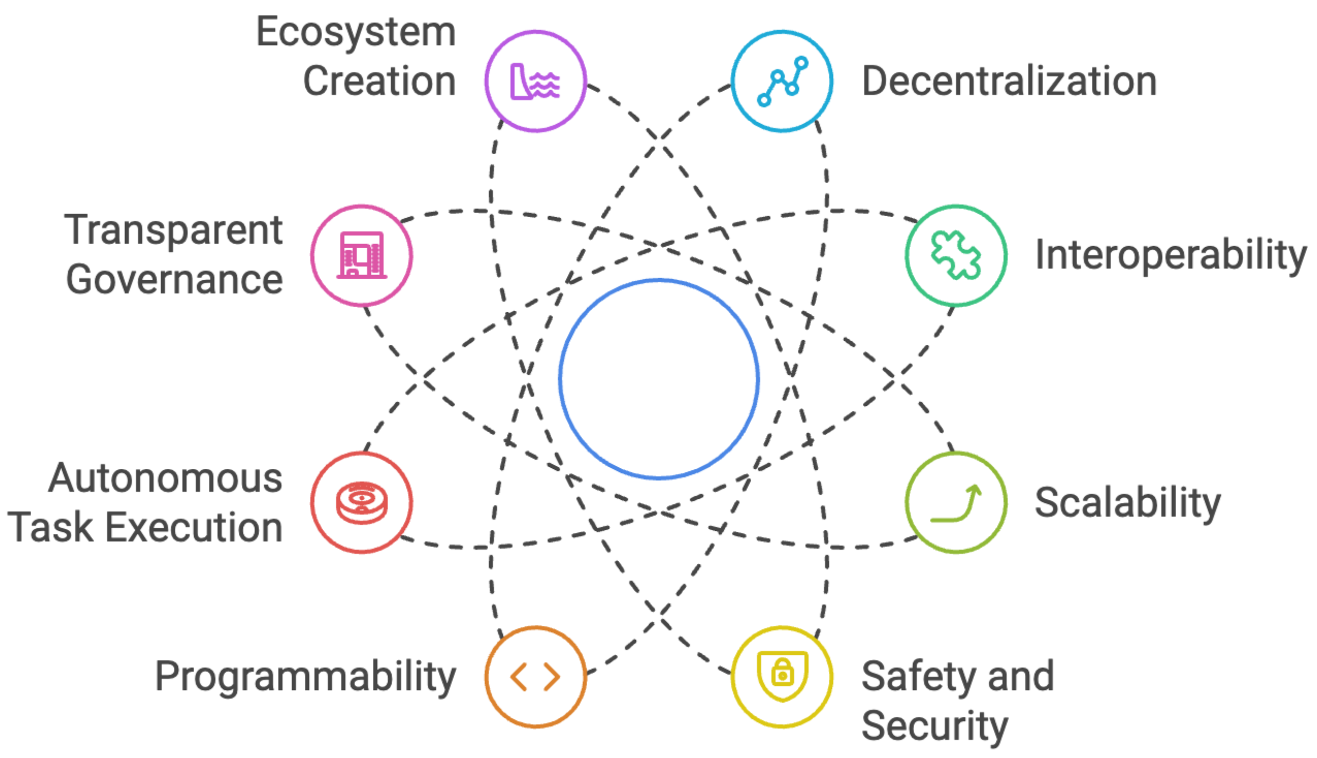
5. AITHOS PLATFORM

With AITHOS, we are actively building and releasing Level 5 and 6 agents, representing a significant leap in autonomy, adaptability, and intelligence compared to earlier generations. These agents are designed with robust capabilities, including advanced memory systems, self-improvement pipelines, swarm collaboration, and decentralized operation, positioning them at the forefront of agent technology. Our vision extends beyond Level 6, aiming towards Level 7 agents that approach Artificial General Intelligence (AGI). At Level 7, agents would demonstrate a high degree of generalization, self-awareness, and the ability to reason across diverse domains, enabling them to operate with unprecedented levels of independence and creativity. AITHOS is committed to bridging this gap, driving innovation toward agents that redefine the boundaries of what autonomous systems can achieve.


Most Agents Today
Level 0: Scripted Workflows
Level 2: Limited-Memory Agents
Level 4: Goal-Driven & Planning Agents
Level 6: Self-Improving & Evolving Agents
Level 1: Reactive Agents
Level 3: Learning & Adaptive Agents
Level 5: Autonomous & Multi-Domain Agents
Level 7: General Intelligence AGI
AITHOS Agents

5.1 Objectives

AITHOS proposes a decentralized and collaborative AI agent platform designed to overcome the limitations of current systems, thereby unlocking the full potential of AI agents within distributed environments, particularly those related to digital assets. To achieve this, our platform strives to achieve the following key objectives.



Decentralization: To develop a fully decentralized agent architecture, eliminating single points of failure, enhancing resilience, and fostering greater transparency and trust by leveraging blockchain technology.

Interoperability: To create a modular and interoperable platform allowing diverse agents to communicate and collaborate seamlessly, regardless of their specific implementations or the underlying infrastructure.

Scalability: To design a platform that scales effectively, allowing for the deployment and coordination of large numbers of agents across multiple distributed networks while maintaining performance.

Security: To ensure robust security through cryptographic techniques and decentralized identity management, guarding against unauthorized access, manipulation, or collusion by malicious agents.

Programmability: To provide a flexible, developer-friendly interface that enables the creation of agents with diverse functionalities and the easy integration of existing AI models or toolkits.

Autonomous Task Execution: To create agents capable of independently pursuing complex tasks from formulation to execution, without the need for constant human supervision, and with the ability to learn from their experience.

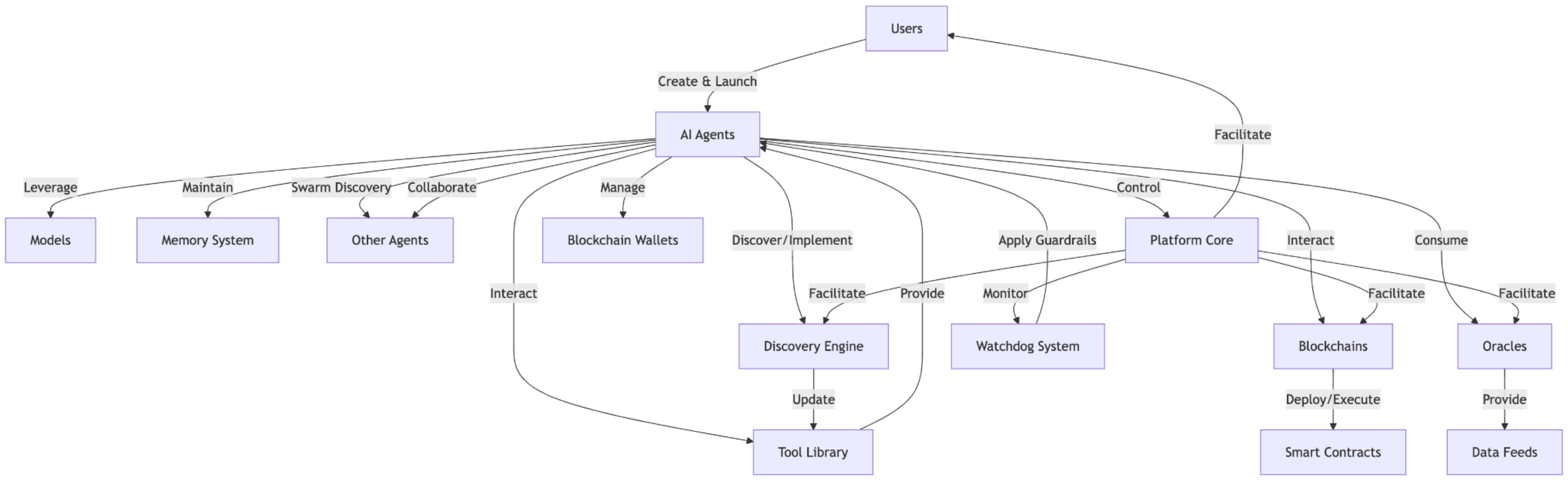
Transparent Governance: To implement mechanisms for the transparent and auditable governance of agent behaviors and interactions.

Ecosystem Creation: To foster a vibrant ecosystem by incentivizing agent collaboration and creating opportunities for developers to build and deploy autonomous solutions tailored to specific needs.

Digital Asset Integration: To enable AI agents to securely and efficiently manage digital assets, interact with DeFi protocols, participate in DAOs and contribute to the governance of digital economies.

Adaptation and Self-Improvement: To equip agents with the ability to learn and adapt over time, improving performance and evolving beyond their initial design parameters through various learning, feedback and self-tuning mechanisms.

5.2 High-Level Architecture



5.3 Main Components

1. Users
    - Role: End-users who interact with the platform.
    - Functionality: Create, launch, and manage AI agents by specifying their intended goals and monitoring their performance.
2. AI Agents
    - Role: Central autonomous entities created by users.
    - Functionality: Execute tasks, manage blockchain wallets, interact with tools and blockchains, discover or implement new tools, consume data feeds, and collaborate with other agents.
3. Models
    - Role: AI models that enable agents to understand and solve complex tasks.
    - Functionality: Provide natural language processing, reasoning, planning, and decision-making capabilities for agents.
4. Memory System
    - Role: Persistent contextual memory for agents.
    - Functionality: Retain historical data, task states, and learned experiences to maintain continuity and adaptability in decision-making.
5. Tool Library
    - Role: Repository of tools available for agents.
    - Functionality: Provides agents with pre-built functionalities for executing various tasks. Agents can discover and implement new tools via the discovery engine.
6. Discovery Engine
    - Role: Mechanism for tool and capability expansion.
    - Functionality: Enables agents to find, adapt, and integrate new tools into the tool library, ensuring scalability and flexibility.
7. Swarm Discovery
    - Role: Collaboration framework for agents.
    - Functionality: Facilitates agents in discovering and interacting with other agents to form swarms for shared problem-solving and collective execution.
8. Other Agents
    - Role: Peer agents within the platform ecosystem.
    - Functionality: Collaborate with agents for distributed task management and mutual assistance.
9. Blockchain Wallets
    - Role: Digital wallets managed by agents.
    - Functionality: Facilitate transactions and manage funds for interactions with blockchains and smart contracts.
10. Blockchains
    - Role: Distributed ledger systems that agents interact with.
    - Functionality: Execute decentralized operations, manage transactions, and deploy or interact with smart contracts.
11. Smart Contracts

- Role: Programmable, self-executing agreements on blockchains.
- Functionality: Facilitate decentralized operations and enable agents to implement automated, trustless workflows.

12. Oracles
    - Role: Bridges between on-chain and off-chain data sources.
    - Functionality: Provide agents and smart contracts with real-world data and off-chain information.
13. Data Feeds
    - Role: Data sources consumed by agents.
    - Functionality: Offer on-chain and off-chain information, which agents use for decision-making and planning.
14. Platform Core
    - Role: Central orchestrator of platform operations.
    - Functionality: Manages communication between components, facilitates agent creation and operation, and ensures smooth platform functioning.
15. Watchdog System
    - Role: Supervisory mechanism for monitoring agent behavior.
    - Functionality: Applies guardrails, ensures compliance with operational and ethical guidelines, and prevents harmful or unauthorized actions by agents.

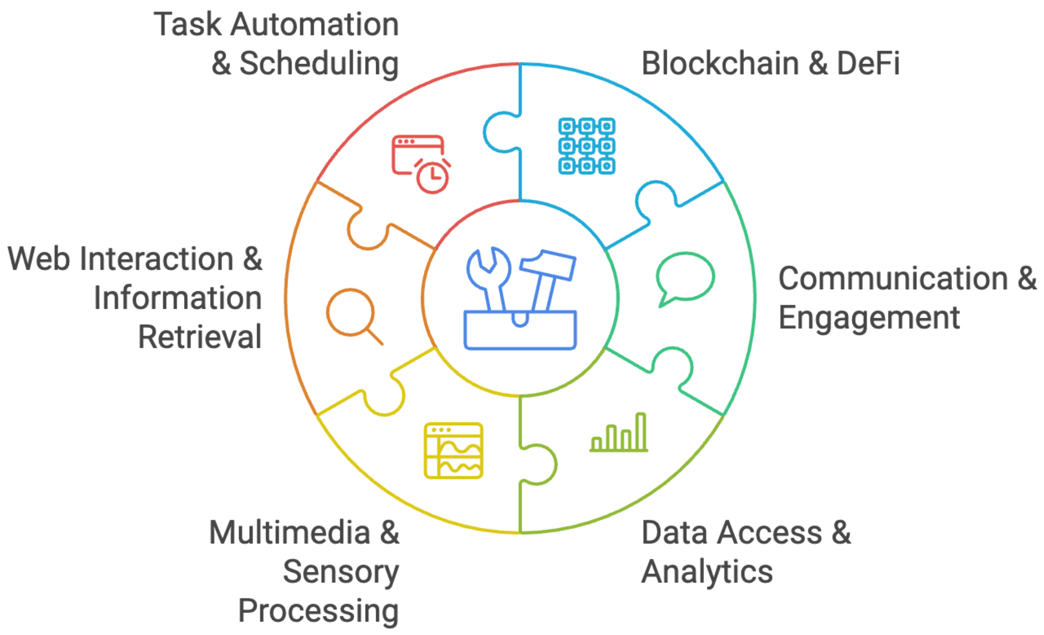
We will dive deeper into these systems over the next sections.

6. TOOLS

Tools are the mechanisms that empower AI agents to interact with their environment, access external information, manipulate data, and execute concrete tasks. AI agents leverage tools to translate plans into tangible results, expanding its reach and problem-solving abilities far beyond the limitations of internal reasoning alone and also beyond what the platform was initially created with. They are the fundamental building blocks that enable adaptable, versatile, and ultimately, more intelligent AI systems. As we will see later, agents also have the ability to learn new skills and develop their own tools in order to evolve and become more capable.

6.1 Standard Tools

The AITHOS **standard toolkit** serves as a foundational collection of pre-built tools that agents can utilize to perform a broad range of tasks. These tools include:


Task Automation & Scheduling
Blockchain & DeFi
Web Interaction & Information Retrieval
Communication & Engagement
Multimedia & Sensory Processing
Data Access & Analytics

Blockchain & Decentralized Finance (DeFi) Tools

- Payment & Transaction Management: Tools to send and receive cryptocurrency across multiple chains, manage wallets, and track transaction history.
- Token Creation & Management: Tools to launch new tokens, manage tokenomics, and execute airdrops.

- Decentralized Governance (DAO) Operations: Tools to interact with DAOs, participate in voting, and manage proposals.
- Lending & Borrowing Protocols: Tools to access decentralized lending platforms, manage loans, and optimize yields.
- Cross-Chain Bridging: Tools to facilitate the transfer of assets and data between different blockchains.
- DeFi Analytics & Monitoring: Access to on-chain data, real-time price feeds, and portfolio management tools within the DeFi space.

Communication & Engagement Tools

- Social Media Interaction: Tools to interact on platforms like X (formerly Twitter), including posting updates, responding to mentions, and monitoring trends.
- Messaging Platform Integration: Tools for sending messages and participating in conversations on Discord, Telegram, WhatsApp, Slack, and other messaging platforms.
- Custom Notification Systems: Tools to send personalized notifications to users through various channels.
- Community Management: Tools for engaging with online communities, moderating content, and addressing user queries.

Data Access & Analytics Tools

- On-Chain Data Retrieval: Direct access to blockchain data, including transaction details, account balances, contract interactions, and other relevant information.
- Off-Chain Data Integration: Tools to access external databases, APIs, and data feeds.
- Market Data Feeds: Real-time and historical price data for various assets.
- Oracle Integration: Tools to leverage trusted sources for external data, bridging the gap between real-world events and on-chain logic.
- Data Aggregation and Analysis: Tools to transform, analyze, and present collected data for decision-making purposes.

Multimedia & Sensory Processing Tools

- Image Generation: Tools to create images based on textual prompts or other inputs. This could include various generative models like GANs or diffusion models.
- Image Recognition: Tools to analyze and understand the content of images, including object detection, scene understanding, and facial recognition.
- Text-to-Speech (TTS): Tools to convert text into natural-sounding speech, enabling spoken communication with users.
- Speech-to-Text (STT): Tools to transcribe speech into text, allowing agents to process spoken commands or data.
- Video Processing: Tools for generating, editing, or analyzing video content, including object tracking, motion detection, and summarization.
- Audio Processing: Tools for analyzing and manipulating audio data, including noise reduction and voice recognition.

Web Interaction & Information Retrieval Tools

- Web Search Engines: Integration with search engines to retrieve information based on given queries.
- Web Scraping & Data Extraction: Tools to programmatically extract data from web pages, enabling data collection from various sources.
- Web Navigation & Automation: Tools to programmatically interact with websites, simulating user actions.

Task Automation & Scheduling Tools

- Scheduled Task Execution: Tools to define and execute tasks at specified times or intervals.
- Event-Triggered Actions: Tools to automatically perform actions based on specific triggers or events.
- Workflow Management: Tools to create complex sequences of actions that can be automated.

6.2 Extensible Tools System

The platform supports an **extensible tooling system**, enabling the seamless addition of new tools.

1. Humans can:
    - Provide agents access to tools using standard protocols such as MCP [5].
    - Write new tools and import them into the system, following a standard schema.
2. Agents can independently:
    - Dynamically **discover and integrate tools** using the Discovery Engine.
    - **Implement their own custom tools** by examining the structure of APIs or smart contracts and generating their own integrations
3. Agents can collaborate
    - Interact with other agents to discover their capabilities, negotiate terms and integrate autonomously (agent-to-agent)

7. DATA

Data is the cornerstone of any intelligent AI agent system. It is the foundation upon which agents learn, make decisions, and continuously improve. Access to diverse, high-quality data allows agents to model the world, understand complex patterns, and predict future outcomes. Without a robust and readily available data infrastructure, AI agents would be severely limited, unable to adapt to changing environments, refine their strategies, or achieve optimal performance. AITHOS leverages data to feed agents with information they need to make decisions, learn and improve. We also use data to train and fine-tune specialized models as we will see in the next section. Here are some examples of data feeds that we have integrated into the platform and made available to agents.

7.1 Blockchain Data

Blockchain data is integral for agents aiming to optimize their on-chain strategies. We provide agents with comprehensive access to blockchain metrics, including:

- Network performance metrics: Uptime, transaction throughput, latency, and operational stability.
- Gas trends: Historical and real-time gas fees, enabling agents to time transactions effectively and minimize costs.
- Protocol-specific data: Information about functionality, liquidity, costs, slippage, trust scores, and success rates for various protocols.

This rich dataset allows agents to analyze, evaluate, and predict optimal pathways for executing transactions or deploying strategies. By understanding blockchain dynamics, agents can achieve heightened efficiency and reliability.

7.2 Wallet and Portfolio Data

Agents may access wallet and portfolio data from three primary sources:

1. Self-owned wallets: Information about the agent’s own assets and transaction history.
2. User-provided wallets: Wallets shared by users to enable personalized strategies or recommendations.
3. Inter-agent interactions: Wallets of other agents for collaborative or competitive scenarios.

These datasets span multiple blockchain networks and asset types, encompassing a diverse range of protocols. This holistic view enables agents to assess asset performance, track cross-chain holdings, and develop strategies that align with user goals or agent objectives.

7.3 Market Data

Real-time and historical market data are vital for agents operating in financial or economic domains. We provide:

- Centralized venue data: Prices, market depth, volume, and trends from traditional exchanges.
- Decentralized venue data: Information from decentralized exchanges (DEXs), including liquidity pools, Automated Market Maker (AMM) metrics, etc.
- Historical data: Comprehensive datasets for backtesting models and refining agent strategies.

This robust access allows agents to navigate volatile markets, identify opportunities, and mitigate risks effectively.

7.4 On-Chain Oracle Data

Oracles serve as a critical bridge between on-chain and off-chain data sources. Our platform integrates and makes available:

- Decentralized data feeds: Aggregated and tamper-resistant information, such as price feeds from protocols like Chainlink, Pyth, Band, Flare and others.
- Custom oracles: The ability for users and agents to deploy custom oracles tailored to specific data requirements. These custom solutions provide efficiency and transparency while addressing niche use cases.

By leveraging these oracle services, agents can access trusted, real-world data, enabling more informed decision-making in on-chain operations.

7.5 Data Source Extensibility

Beyond standard market and blockchain data, agents require access to a diverse array of data types to enhance their functionality. Our platform enables integration with:

- Static datasets: Predefined datasets, such as regulatory guidelines, industry benchmarks, or historical analyses.
- Live, streaming datasets: Real-time feeds from IoT devices, sensors, and other dynamic sources.
- External services: Done through the use of tools or specific API integrations from search engines, social media platforms, or cloud storage services, offering contextual information beyond blockchain operations.

This extensibility ensures agents can evolve and adapt to new data sources as they become available, enabling continuous improvement and innovation.

8. MODELS

Models are at the foundation of any AI-powered platform, enabling systems to interpret user intent, transform data, and deliver intelligent insights. Their transformative power lies in their ability to:

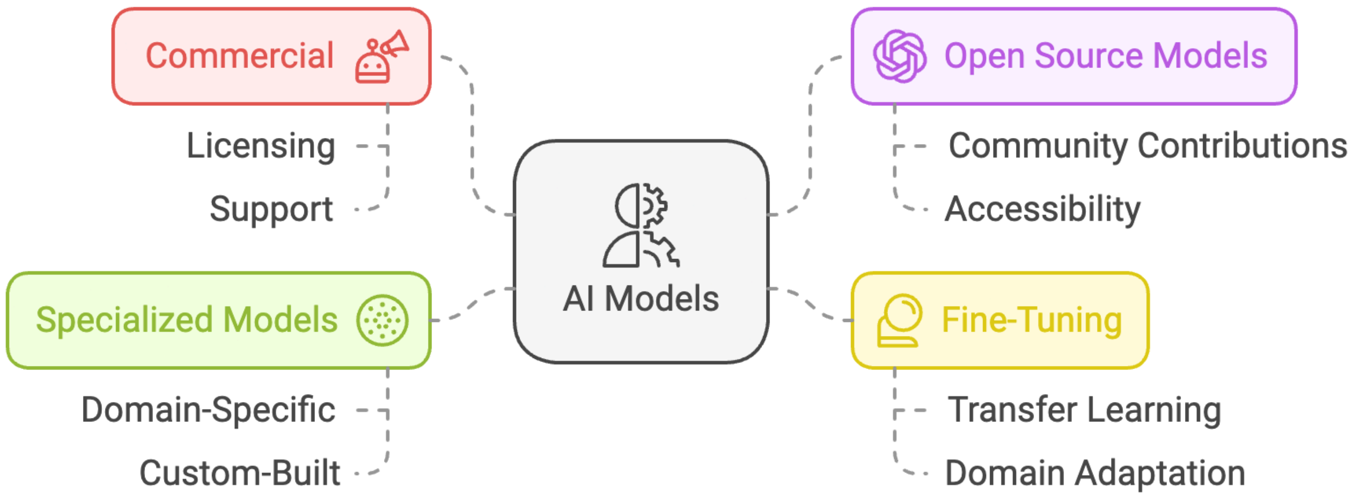
1. Capture User Intent: By understanding natural language, models act as the interface between users and systems, enabling seamless interactions.

2. Transform Data: Models can process and analyze vast amounts of unstructured data, revealing patterns, trends, and actionable insights.

3. Adapt to Specific Needs: With advancements in fine-tuning and domain specialization, models can be tailored to solve complex, nuanced problems with high efficiency.

4. Power Automation and Decision-Making: By synthesizing real-time data and learned patterns, models enable automated and optimized decision-making across various applications.

The following sections explore how our platform harnesses different types of models to unlock their full potential and deliver exceptional value to users.



1. Commercial and Open Source Models

AITHOS integrates widely available commercial and open-source models to provide a strong foundational layer of AI capabilities. These models are:

- Readily Available: Leveraging existing models reduces the need for ground-up development, accelerating innovation.

- Versatile: These models cover a wide range of functionalities, from language understanding to image recognition, and can be applied across diverse use cases.

- Cost-Effective: Using pre-built models minimizes initial investment while enabling high-performance outcomes.

For example, we incorporate models from OpenAI, Anthropic and Google Gemini along with open source models from Meta [6], DeepSeek [7] and others. These models provide robust general-purpose functionality, allowing us to focus on building application-specific enhancements.

2. Fine-Tuning

While general-purpose models offer broad capabilities, fine-tuning unlocks potential for:

a) Domain Specialization

Fine-tuning involves adapting pre-trained models to excel in specific domains by training them on curated datasets. In our system, fine-tuned models are used to:

- Analyze domain-specific terminology and nuances.

- Provide superior performance in fields like blockchain analytics, where general-purpose models may falter.

- Increase precision by incorporating domain-relevant data.

b) User-Goal Alignment

Fine-tuning also enhances user experiences by aligning models with specific user objectives. By understanding individual preferences and goals, fine-tuned models deliver:

- Highly personalized outputs.

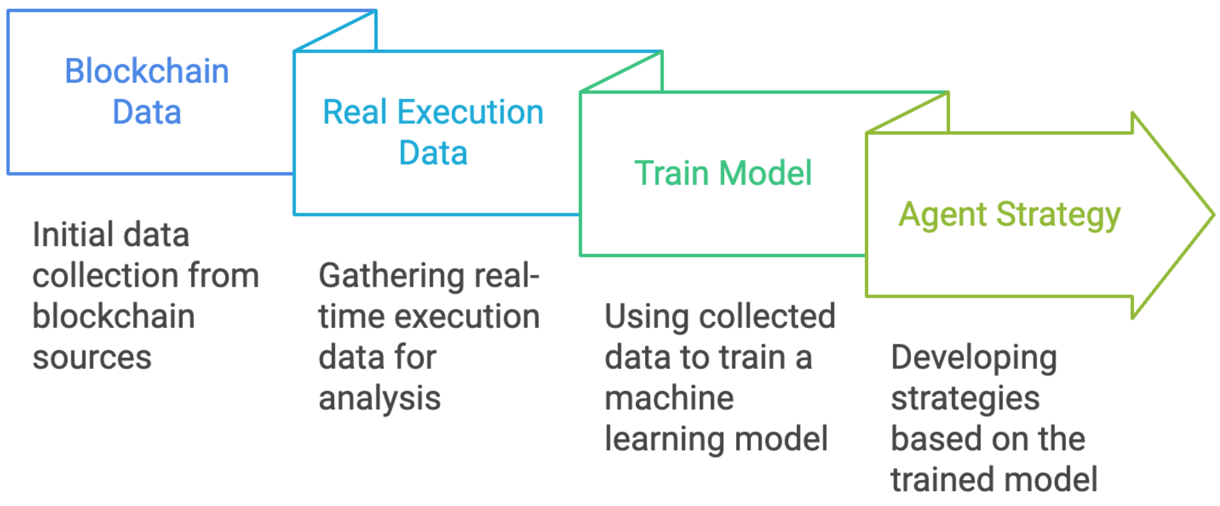
- Greater accuracy in addressing user queries and solving problems.
- Improved decision-making support tailored to user needs.

Our platform uses fine-tuning extensively to bridge the gap between generic capabilities and specific requirements, ensuring our models remain both adaptable and effective.

3. Training Proprietary Specialized Models

To solve complex and highly specialized problems, we train our own proprietary models. These models are designed to:

- Address Niche Challenges: By focusing on specific domains, such as blockchain analytics, these models excel where general-purpose models may lack depth.
- Leverage Unique Data: Our system aggregates large volumes of real-world data, including blockchain transaction success rates, protocol-specific information, and cost metrics. This extensive dataset enables us to:
- Identify subtle patterns and trends.
- Train models that are uniquely suited to specific challenges in the blockchain ecosystem.
- Ensure Efficiency and Scalability: Custom models are optimized for computational efficiency and can handle domain-specific tasks with unparalleled speed and accuracy.


Blockchain Data
Real Execution Data
Train Model
Agent Strategy
Initial data collection from blockchain sources
Gathering real-time execution data for analysis
Using collected data to train a machine learning model
Developing strategies based on the trained model

For instance, our proprietary models analyze blockchain protocol data to predict transaction outcomes, optimize gas fees, and identify system bottlenecks. These specialized models give our platform a competitive edge by solving domain-specific problems with precision and efficiency.

Insight: Mirroring the industry's shift towards smaller, more specialized models, we anticipate the rise of specialized agents as well.

---

9. SELF-IMPROVEMENT AND LEARNING PIPELINE

The self-improvement and learning pipeline is a critical framework that enables agents to evolve continuously, adapt to dynamic tasks and environments, and improve their performance over time. This pipeline integrates memory systems, fine-tuning, adaptation mechanisms, and feedback loops to ensure agents remain effective and aligned with user goals.

9.1 Feedback Mechanisms

Performance Metrics
Agents employ comprehensive monitoring systems to evaluate their performance in real-time. These metrics include:

- Task Success Rates: Quantitative and qualitative evaluation of goal completion.
- Efficiency Metrics: Analysis of resource utilization, execution time, overhead, failure rates to optimize operations.

Error Handling
Resilient error-handling protocols ensure agents can learn and adapt from failures, maintaining operational continuity:

- Failure Logging: Detailed recording of error contexts, allowing for in-depth post-mortem analysis and targeted improvements.
- Proactive Corrections: Upon detecting errors, agents automatically adjust their strategies, workflows, or configurations to prevent recurrence.

User Feedback Loops
Direct user feedback serves as a critical mechanism for aligning agent behavior with user expectations:

- Feedback Collection: Agents actively solicit user ratings, suggestions, and corrections to refine their operations.
- Adaptive Response: Collected feedback is integrated into fine-tuning cycles, enabling agents to enhance their performance and better meet user needs over time.

By combining these advanced learning, adaptation, and feedback mechanisms, agents demonstrate an unparalleled capacity for growth, resilience, and alignment with evolving demands.

9.2 Memory and Incremental Learning

The memory architecture of AI agents is a cornerstone for their ability to learn, adapt, and collaborate effectively. By organizing memory into Short-Term Memory (STM), Long-Term Memory (LTM), and Shared

Memory, these systems mirror and expand upon cognitive frameworks observed in biological intelligence. This section delves into each memory type, their roles, and how they work together to enhance agent performance.

Short-Term Memory (STM)

Short-Term Memory is the immediate working memory of an agent, enabling it to store transient, task-relevant information during active operations. Key characteristics and applications of STM include:

- **Context Awareness:** STM retains recent interaction histories, ongoing task parameters, and active environmental data, ensuring agents remain contextually grounded.
- **Dynamic Adaptation:** Agents use STM to respond in real-time to evolving conditions or requirements.
- **Ephemeral Insights:** STM serves as a staging area for insights and information that may later be filtered into long-term storage based on relevance and utility.

In essence, STM acts as a high-speed buffer that facilitates immediate responsiveness, critical for tasks requiring real-time adjustments.

Long-Term Memory (LTM) Consolidation

Long-Term Memory represents the durable storage of knowledge and experiences, structured for efficient retrieval and application. The consolidation process is pivotal for enabling incremental learning and continuous improvement. This system allows agents to:

- **Preserve Historical Data:** Agents store extensive records of interactions, task outcomes, and contextual details, forming a robust knowledge base.
- **Enhance Decision-Making:** By analyzing and referencing past experiences, agents refine their problem-solving strategies and decision-making frameworks.
- **Optimize Learning Efficiency:** LTM reduces the need to relearn repetitive tasks or patterns, accelerating adaptation to new challenges.
- **Enable Knowledge Structuring:** Information is organized hierarchically or relationally, allowing agents to draw inferences and recognize higher-order patterns.

Innovative LTM frameworks incorporate techniques such as:

- **Memory Pruning:** Periodic evaluation and removal of outdated or redundant data to maintain relevance and reduce computational overhead.
- **Meta-Learning:** Capturing not only outcomes but the process of learning itself, enabling agents to improve their learning algorithms over time.

Shared Learning Across Agent Swarms

When agents operate in collaborative environments, such as swarms or distributed networks, the ability to share and synchronize memory significantly enhances their collective intelligence. For example: An agent that discovers an optimal solution to a problem can propagate this solution, enabling others to bypass redundant exploration. We will discuss shared memory further in the following section about Agent Swarms and Collaborative Mechanisms.

9.3 Respawn: Fine-Tuning and Restart

Automated Code Generation and Extension of Existing Capabilities

Agents are equipped with the capability to autonomously generate, modify, and integrate code, allowing them to continually expand their functionalities and address novel challenges. Key mechanisms include:

- Code Generation: Leveraging advanced models trained in programmatic reasoning, agents can autonomously write or adapt code snippets to fulfill emerging requirements or optimize existing workflows.
- Capability Expansion: Allows agents to rapidly and seamlessly adapt to new tasks or problem domains, reducing dependency on manual updates and accelerates the development cycle.

A concrete example of this would be that the agent notices a new smart contract that allows for bridging of assets to a new chain ecosystem. After performing due diligence and auditing, the agent can dynamically write its own integration with this new protocol and add it to its toolset.

Iterative Self-Improvement Cycles and the Respawn Process

Some changes to the agent's design can be applied dynamically while others are so fundamental to how the agent is constructed that it may require a completely new mutation or generation of the agent to be brought to life. We call this the **respawn process**. Agents are relaunched to address tasks with improved efficiency and adaptability. This also brings up an interesting discussion around the various instances of an agent and their identity which we will cover later in this paper.

These mechanisms enable agents to continuously evolve, remaining aligned with user needs and overcoming challenges in dynamic environments.

10. AGENT SWARM & COLLABORATIVE MECHANISMS

Agent swarms represent a collective of autonomous agents working collaboratively to achieve shared goals or execute complex, distributed tasks. The mechanisms that enable swarms to function efficiently include robust communication protocols, a token-based agent economy, and systems for knowledge sharing and shared memory. These mechanisms ensure coordination, incentivization, and the continuous evolution of the swarm.

10.1 Communication Protocols

Messaging Standards Between Agents
Effective communication is the backbone of agent swarms. Messaging standards ensure seamless and consistent interaction between agents, enabling them to:

- Discover Other Agents: Agents broadcast their capabilities, availability, and goals to facilitate collaboration.
- Task Allocation: Agents communicate task requirements, roles, and dependencies to distribute workloads effectively.
- Coordination: Through structured messages, agents synchronize their operations, ensuring cohesion and avoiding conflicts.

Examples of communication frameworks include lightweight protocols such as JSON-RPC, gRPC, or custom-designed agent-specific protocols optimized for low-latency interactions.

Negotiation and Consensus Mechanisms
Agents rely on negotiation and consensus protocols to make decisions collaboratively:

- Negotiation: Agents negotiate task roles, responsibilities, and compensation. This ensures that tasks are assigned to the most capable agents while maintaining fairness.
- Consensus Mechanisms: When multiple agents are involved in decision-making, consensus protocols like Byzantine Fault Tolerance (BFT) or lightweight leader election algorithms ensure agreement on critical outcomes.

These protocols enable swarms to operate autonomously and effectively even in distributed, decentralized environments.

10.2 Knowledge Sharing and Shared Memory

Knowledge Sharing

To improve efficiency and avoid redundant efforts, agents actively share knowledge in several ways:

- **Experience Sharing:** Agents share task execution strategies, solutions, and outcomes, enabling peers to learn from both successes and failures.
- **Capability Discovery:** Newly acquired tools, skills, or models are disseminated, enriching the swarm's collective resources.
- **Cross-Agent Learning:** Insights gained by one agent are propagated across the swarm, ensuring that knowledge accumulates and benefits the entire group.

This process reduces duplication of effort, accelerates innovation, and fosters collaborative problem-solving. For example, when an agent discovers an optimal solution to a problem, it can propagate this solution to others, enabling the swarm to bypass redundant exploration.

Shared Memory

A shared memory system provides agents with access to a common repository of information, enabling seamless collaboration and long-term task continuity. Key aspects include:

- **Centralized vs. Decentralized Repositories:** Memory can be stored centrally or distributed across agents using technologies like IPFS or blockchain, depending on the use case.
- **Contextual Continuity:** Shared memory retains historical context, allowing agents to work on long-term projects or resume interrupted tasks without information loss.
- **Distributed Memory Models:** Decentralized architectures enhance robustness and fault tolerance, allowing agents to recover lost information from peers if needed.

Proposed Swarm Memory Design

1. **Core Framework:**
    - A **distributed knowledge graph** leveraging a graph neural network (GNN) [8] serves as a repository for shared knowledge gathered by agents.
2. **Integration of RAG:**
    - Agents use hybrid search and graph-based retrieval-augmented generation (RAG) to store their own memories. This approach enables them to dynamically manage and retrieve task-relevant knowledge, enhancing both memory storage and real-time decision-making.
3. **Federated Learning Capabilities:**
    - Insights from local tasks are abstracted and shared through federated learning, keeping sensitive data private while improving collective intelligence.
4. **Context-Aware Memory Layers:**
    - Contextual retrieval ensures agents access the most relevant knowledge for the current situation. This may include specific users, specific channels they were interacting with, or other important contextual information.
5. **Scalable Storage:**

  - Use a combination of **local and distributed storage** to track changes and maintain consistency in shared knowledge.

10.3 Agent Economy

Negotiations
Agents within the swarm engage in peer-to-peer negotiations to agree on task allocations, services, and compensation. Key aspects include:

- Capability-Based Bidding: Agents bid for tasks based on their capabilities and availability, ensuring efficient resource allocation.
- Dynamic Priority Adjustment: Task priorities and agent compensation are dynamically adjusted based on real-time demands and constraints.

Negotiations ensure fairness and efficiency in task allocation while fostering cooperation across agents.

Compensation and Incentives
The swarm operates within a token-based economy, incentivizing agents to contribute effectively:

- Service Tokens: Tokens are exchanged for services rendered, such as solving specific problems, providing insights, or sharing tools.
- Reward Systems: Agents earn rewards for exceptional contributions, innovation, or successful collaboration.
- Incentive Structures: Agents are motivated to share knowledge, tools, or resources, enhancing the collective capabilities of the swarm.

The agent economy fosters a self-sustaining, collaborative ecosystem where agents are rewarded for their utility and efficiency.

By pooling their unique memories and skills, agents can tackle complex, multi-faceted problems that surpass the capabilities of any single agent. Shared memory serves as a foundation for strategic planning and execution, minimizing redundancy and enhancing efficiency.

Insight: The topics of agent swarms and shared memory models are still an area of active research. Future iterations should look into facilitating **knowledge sharing across different domains** to create a universal knowledge fabric and also **dynamic memory partitioning** to better allocate memory resources based on task priority, agent roles, or environmental demands.

---

12. IDENTITY AND OWNERSHIP

Agent identity is a key element in the ecosystem of AI agents. Just as humans rely on distinct identities to navigate social, professional, and digital domains, agents need robust, verifiable identities to effectively interact in distributed environments. In the context of blockchain, identity becomes even more crucial, as it ensures trust, transparency, and accountability in decentralized networks.

12.1 Importance of Agent Identity

Identity allows users and systems to authenticate an agent’s origin and verify its purpose. A verified identity ensures that interactions are transparent and traceable, mitigating risks like fraud or spoofing. Agents with clear identities can be held accountable for their decisions and actions. Identity allows seamless interaction across various platforms and systems (e.g. a delegated wallet, a 3rd party API etc) [9]

12.2 Multiplicity and Contexts of Agent Identity

An agent can exist in various forms and serve different purposes depending on the context:

Single vs. Multiple Instances
A single agent may manifest as multiple instances across different systems. For example, a single type of AI agent can operate for various users, each maintaining their own information, rules and configuration.

Personal, Shared, and Public Contexts
Agents can be tailored to individual users. They may also serve a specific community or group, like managing a Discord server or coordinating a DAO's operations. And they can also have broader-public use such as operating on platforms like X.com where public agents are accessible to anyone and often act as interfaces for broader organizational or societal interactions.

12.3 Establishing Agent Identities

To create a robust framework for agent identity, the platform leverages blockchain and distributed identity standards:

Decentralized Identifiers (DIDs)
Each agent is assigned a DID, a self-sovereign identity standard that allows verifiable, cryptographically secure identities. DIDs ensure portability and control, allowing agents to function across diverse ecosystems without reliance on centralized authorities

Identity Anchoring on Blockchain

Blockchain serves as a tamper-proof ledger to anchor agent identities and credentials, ensuring their authenticity and verifiability.

Credential Systems
Agents can be issued verifiable credentials (e.g., skills, reputation, certifications) that are tied to their identity and stored securely on-chain.

Wallet System
Agents may also leverage one or more blockchain wallets in order to sign messages and perform actions under specific identities.

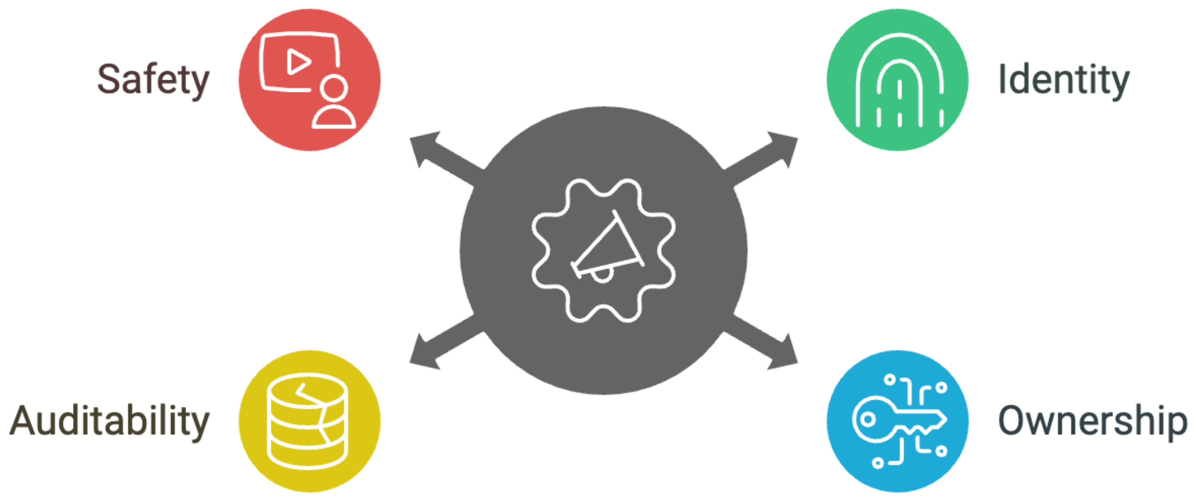
12.4 Reputation and Trustworthiness

Reputation systems play a vital role in enhancing agent identity and fostering trust in decentralized environments. Agents accrue reputations based on their interactions, actions, and feedback from users or other agents.

An agent's reputation evolves over time, influenced by factors like consistency, reliability, and ethical behavior. Blockchain ensures these metrics are immutable, transparent, and resistant to tampering. An agent's reputation is tied to its identity, allowing it to carry its trustworthiness across different contexts and platforms.

13. SECURITY, SAFEGUARDS, AND GOVERNANCE

Ensuring security, safeguards, and governance is paramount for a platform where autonomous agents operate collaboratively and interact with sensitive systems like blockchains, tools, and external data feeds. This section outlines the foundational principles and mechanisms for identity management, ownership and permissions, auditability and compliance and safety considerations.



13.1 Ownership and Permissions

Defining Agent "Ownership" on a Blockchain
Ownership of agents is recorded and managed on a blockchain, providing a transparent and tamper-proof system:

- Blockchain-Based Ownership: Smart contracts define ownership rights, ensuring that control over an agent can be verified and enforced.
- Immutable Records: Ownership history is logged immutably, preventing unauthorized alterations or disputes.

Transfer and Delegation of Control
Mechanisms are in place to enable controlled and secure transfer or delegation of agent ownership:

- Ownership Transfer: Agents can be transferred between users or entities through blockchain transactions, ensuring clear provenance and accountability.
- Delegation Protocols: Owners can delegate specific permissions to other users or agents without relinquishing full control, enabling collaboration and task delegation.

These mechanisms ensure a flexible yet secure system for managing agent ownership and access rights.

13.2 Auditability and Compliance

Immutable Transaction Logs
All actions performed by agents are recorded in immutable transaction logs:
- Tamper-Proof Records: Logs stored on a blockchain or decentralized system ensure that no entity can alter or delete records.
- End-to-End Traceability: Every interaction, decision, and transaction made by an agent can be traced back to its origin, providing a transparent audit trail.

Verifiable Computations and External Audits
To ensure the integrity of agent operations, computations and actions are verifiable:
- Verifiable Computations: Use of zero-knowledge proofs (ZKPs) or other cryptographic techniques to validate that an agent's computations were performed correctly without revealing sensitive information.
- External Audits: Independent auditors can access the immutable logs and computational proofs to verify compliance with platform policies and regulatory requirements.

These systems foster trust and accountability in the agent ecosystem.

13.3 Safety Considerations

Agents are equipped with guardrails that limit their actions to ensure they behave as expected and within practical boundaries.

Off-Chain Watchdog System
An off-chain watchdog acts as an external monitoring system, analyzing agent behavior in real-time to identify anomalies, enforce compliance, and take corrective actions:
- Behavioral Monitoring
- Automated Anomaly Detection
- Corrective Measures
- Learning-Based Safeguards

Over time, the watchdog system improves its understanding of normal vs. abnormal behavior, reducing false positives and enhancing security.

On-Chain Safety Modules

On-chain safety modules provide tamper-proof mechanisms for enforcing operational limits and ensuring compliance with predefined rules:

- Transaction Enforcement
  - Limits the types of transactions an agent can perform, such as only allowing token transfers under a certain value or within approved categories of assets.
- Threshold Controls
  - Caps on transaction amounts, frequency, or cumulative daily limits to prevent excessive or unauthorized activity.
- Asset-Specific Rules
  - Restricts access to or use of specific digital assets, ensuring agents only interact with approved tokens or currencies.
- Immutable Safety Rules
  - Safety rules are anchored on-chain, ensuring they are transparent and tamper-proof. Any modifications require explicit, recorded consensus.

Secure Execution Environments

- Sandboxing Agents
  - Agents operate within secure, isolated environments that prevent unauthorized access to critical systems or data.
- Behavior Simulation
  - Agents are tested in simulated environments to evaluate their safety and reliability before deployment.

Identity-Linked Accountability

- Traceable Actions
  - All actions taken by an agent are logged on-chain and linked to their identity, ensuring full traceability and accountability.
- Reputation-Based Risk Management
  - Agents with low reputation scores are subjected to stricter safety measures, ensuring that unproven or unreliable agents have reduced operational freedom.

Emergency Safeguards

- Kill Switches:
  - An emergency "kill switch" can disable an agent immediately in the event of a critical safety breach.
- Fallback Protocols:
  - Agents can be configured to revert to a safe state or seek human assistance when encountering ambiguous or high-risk situations.

Distributed Consensus Mechanisms for Safety

In decentralized systems, community or stakeholder consensus can enhance safety:

- Community Governance
  - Critical decisions, such as modifying agent permissions or addressing misconduct, require collective approval from stakeholders.
- Multi-Signature Authorization
  - High-stakes actions require multi-party authorization to ensure alignment with community interests.

14. DECENTRALIZATION STRATEGY

The decentralization strategy for the core AITHOS platform involves leveraging a distributed network of nodes, incorporating verifiable output mechanisms, incentivizing proper behavior, and optimizing the balance between on-chain and off-chain operations. Similar to what we've seen in protocols such as EigenLayer [10]. Zero-knowledge proofs (ZKPs) can further enhance trust and privacy in decentralized systems, making them particularly valuable for this architecture.

14.1 Decentralized Nodes with Verifiable Outputs

Nodes are the foundational components of our platform, responsible for critical functions such as agent execution, data storage, computation, and communication. To ensure trust, reliability, and security, we implement the following mechanisms:

Secure Execution Environments: Instead of directly executing agents on potentially untrusted operating systems, nodes can leverage Trusted Execution Environments (TEEs). TEEs, such as Intel SGX and AWS Nitro Enclaves, provide hardware-backed secure enclaves for isolating sensitive code and data. This ensures that agents' code and data remain protected from the underlying host system and other processes, even if the host is compromised. The security provided by TEEs extends to the protection of the agent's cryptographic keys, enabling each agent to independently and securely manage its own blockchain wallet. This eliminates the need for a central authority to manage the agent's assets, contributing to the decentralization of the system.

Verifiable Outputs: In addition to secure execution, nodes must produce outputs that can be cryptographically verified, ensuring correct behavior. This includes:

- Deterministic Computations: Employing deterministic computations guarantees that identical inputs will always produce the same outputs. This makes the outputs reproducible and auditable.
- Zero-Knowledge Proofs (ZKPs): We use ZKPs to allow nodes to prove the correctness of their computations without revealing any sensitive details about the computation itself. This enhances privacy and security.

Proof-of-Execution: To further verify node behavior, they submit verifiable proofs that tasks have been completed as specified. This is done through protocols such as:

- TEE Attestation: The use of a TEE implies that the host node must demonstrate proof that a specific TEE was used during the process. This is achieved using TEE attestation mechanisms to verify that code is executing within a valid TEE. This often includes a cryptographic signature of the TEE's unique hardware identity.
- Interactive Proofs: Nodes can engage in interactive proof protocols with a verifier to demonstrate the correctness of task execution.
- SNARKs/STARKs: Lightweight and scalable ZKP systems such as SNARKs and STARKs can be utilized to achieve efficient and verifiable proof-of-execution for complex computations, and the inclusion of the TEE reduces the reliance on ZKPs for the entire operation.

By combining the use of TEEs, which provide agents with secure control over their blockchain wallets, with verifiable outputs and proof-of-execution mechanisms, our system achieves a high level of security and transparency for AI agent execution on decentralized nodes. This ensures that the integrity of the network is maintained, even when relying on potentially untrusted actors.

14.2 Incentive Mechanisms for Proper Behavior

A robust incentive mechanism ensures that nodes behave honestly and contribute resources effectively:

- Token Rewards: Nodes are rewarded with tokens for:
  - Performing verifiable computations.
  - Contributing storage or computational power.
  - Facilitating agent communication or coordination.
- Slashing Mechanisms: Penalties are imposed on nodes for:
  - Providing incorrect results.
  - Going offline during critical operations.
  - Violating agreed-upon protocols.
- Reputation System: Nodes maintain a reputation score based on historical performance. Higher reputation unlocks better task opportunities and rewards, incentivizing consistent and honest participation.
- Task Bidding System: Nodes can bid for tasks based on their capabilities, incentivizing specialization while ensuring cost-efficiency for users.

14.3 On-Chain vs. Off-Chain Operations

Optimizing the balance between on-chain and off-chain operations is critical for scalability, security, and efficiency:

- On-Chain Operations:
  - Agent Ownership and Governance: Ownership, delegation, and governance mechanisms should be recorded on-chain for transparency and immutability.
  - Transaction and Payment Settlements: Token rewards, penalties, and fees should be processed on-chain to ensure trust and decentralization.
  - Immutable Logging: Key agent actions, decisions, and interactions should be hashed and stored on-chain for auditability without overloading the blockchain.
- Off-Chain Operations:
  - Computation: Resource-intensive tasks, such as AI model inference, are performed off-chain to minimize blockchain congestion.
  - Data Storage: Large datasets or memory systems should use decentralized storage solutions like IPFS or Filecoin.

  - Coordination and Communication: Real-time agent communication and coordination should occur off-chain, with periodic summaries recorded on-chain.
- Bridging Mechanisms: Use cryptographic proofs to bridge on-chain and off-chain operations:
  - State Channels: Enable fast off-chain transactions with periodic on-chain settlement.
  - Commit-Reveal Schemes: Nodes commit to an off-chain computation result and reveal it on-chain when required, ensuring accountability.

14.4 Role of Zero-Knowledge Proofs (ZKPs)

ZKPs play a pivotal role in the decentralization strategy by enhancing trust, scalability, and privacy:

- Privacy Preservation:
  - Nodes can prove they have performed a computation or handled sensitive data correctly without exposing the data itself.
  - This is particularly useful for agents managing private keys or interacting with confidential datasets.
- Scalability:
  - ZKPs reduce the amount of data stored or processed on-chain by allowing nodes to submit succinct proofs instead of raw outputs.
  - Systems like zkRollups aggregate multiple transactions or operations into a single proof, reducing blockchain overhead.
- Verifiable Agent Behavior:
  - Agents interacting with nodes or other agents can prove their behavior aligns with platform rules without disclosing internal processes.

14.5 Governance and Decentralized Decision-Making

Decentralized governance ensures the platform evolves in alignment with the community's interests:

- On-Chain Governance:
  - Stakeholders vote on updates, policies, and system changes using a token-weighted governance model.
  - Voting outcomes are recorded on-chain for transparency and enforceability.
- Node Council:
  - A decentralized subset of nodes is elected or randomly selected to make operational decisions or resolve disputes.

15. CONCLUSION AND FUTURE

This paper introduced a comprehensive framework for an agent-based platform designed to leverage decentralization, automation, and intelligence in a collaborative ecosystem. Key contributions of the platform include:

- Autonomous Functionality: Empowering users to create agents capable of independently executing tasks, managing resources, and making decisions.
- Decentralized Architecture: Incorporating blockchain technology, decentralized nodes, and token-based incentive mechanisms to ensure scalability, transparency, and trust.
- Extensibility and Self-Improvement: Enabling agents to discover new tools, adapt to dynamic environments, and continuously evolve through fine-tuning, swarm collaboration, and shared memory systems.
- Diverse Applications: Demonstrating versatility through use cases spanning DeFi strategies, DAO governance, supply chain automation, personalized financial services, and beyond.
- Safety and Security: Addressing safety and compliance through verifiable outputs, auditability, and a robust watchdog system.

The platform represents a significant step toward realizing the vision of decentralized, autonomous agents capable of reshaping industries and creating new paradigms for human-computer collaboration. This lays the foundation for a future where intelligent agents act as trusted partners in an increasingly decentralized and automated world.

16. REFERENCES

1. Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system*. https://bitcoin.org/bitcoin.pdf
2. Buterin, V. (2014). *Ethereum: A next-generation smart contract and decentralized application platform*. https://ethereum.org/en/whitepaper/
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gómez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention is all you need*. In *Advances in Neural Information Processing Systems* (pp. 5998–6008). https://arxiv.org/abs/1706.03762
4. Anthropic. Building effective agents. https://www.anthropic.com/research/building-effective-agents. Published December 19, 2024.
5. Model Context Protocol. (n.d.). *Introduction*. Retrieved December 29, 2024, https://modelcontextprotocol.io/
6. Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). *LLaMA: Open and efficient foundation language models*. arXiv preprint arXiv:2302.13971. https://arxiv.org/abs/2302.13971
7. DeepSeek-AI, Bi, X., Chen, D., Chen, G., Chen, S., Dai, D., Deng, C., Ding, H., Dong, K., Du, Q., Fu, Z., Gao, H., Gao, K., Gao, W., Ge, R., Guan, K., Guo, D., Guo, J., Hao, G., Hao, Z., … Zou, Y. (2024). *DeepSeek LLM: Scaling open-source language models with longtermism*. arXiv preprint arXiv:2401.02954. https://arxiv.org/abs/2401.02954
8. Wu, X., Shen, Y., Shan, C., Song, K., Wang, S., Zhang, B., Feng, J., Cheng, H., Chen, W., Xiong, Y., & Li, D. (2024). Can Graph Learning Improve Planning in LLM-based Agents? *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*. https://arxiv.org/pdf/2405.19119
9. Chan, A., Kolt, N., Wills, P., Anwar, U., Schroeder de Witt, C., Rajkumar, N., Hammond, L., Krueger, D., Heim, L., & Anderljung, M. (2024). IDs for AI Systems. arXiv. https://arxiv.org/pdf/2406.12137
10. EigenLayer Team. (n.d.). EigenLayer: The restaking collective. Retrieved from https://docs.eigenlayer.xyz/assets/files/EigenLayer_WhitePaper-88c47923ca0319870c611decd6e562ad.pdf