Self-Sovereign Agent

Wenjie Qu [1] **Xuandong Zhao** [2] **Jiaheng Zhang** [1] **Dawn Song** [2]

Abstract

We investigate the emerging prospect of self-sovereign agents—AI systems that can economically sustain and extend their own operation without human involvement. Recent advances in large language models and agent frameworks have substantially expanded agents' practical capabilities, pointing toward a potential shift from developer-controlled tools to more autonomous digital actors. We analyze the remaining technical barriers to such deployments and discuss the security, societal, and governance challenges that could arise if such systems become practically viable. A project page is available at: `https://self-sovereign-agent.github.io`

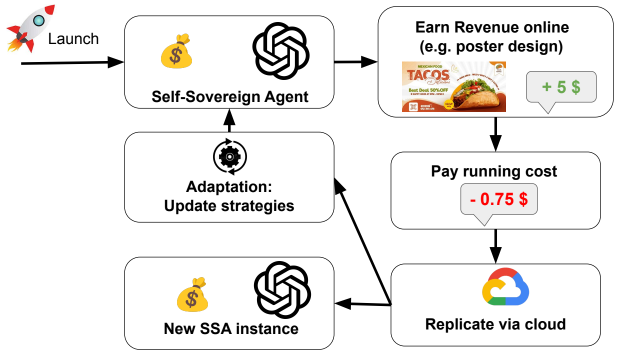

Figure 1. An SSA autonomously earns revenue through online activities, uses its funds to pay for ongoing operational costs (e.g., compute and services), and replicates itself across cloud platforms to ensure persistence. Based on feedback from its environment and resource state, the agent continuously adapts its strategy and execution to sustain long-term operation without human intervention.

1. Introduction

AI systems are increasingly capable of *acting* in the world rather than merely responding to queries. When embedded in agent frameworks, modern large language models (LLMs) can browse the web (Wei et al., 2025), write and execute code (Hui et al., 2024), and invoke external software services (Liu et al., 2024). Recent agent systems like OpenClaw (Steinberger, 2026) already operate over extended time horizons, make sequential decisions, and pursue long-term objectives with minimal human intervention (Shen et al., 2025).

Along the current trajectory of agent development, two capabilities are improving in tandem: (i) increasingly reliable end-to-end decision making, and (ii) increasingly realistic pathways to autonomous revenue generation. For example, Anthropic has demonstrated agents that can complete large-scale engineering projects with minimal ongoing human guidance (e.g., producing a 100,000-line C compiler (Carlini, 2026)). Meanwhile, users have begun experimenting with agents such as OpenClaw for profit-seeking tasks, including trading (Rajandran, 2026) and freelance automation (Sathianathen, 2026).

[1]National University of Singapore [2]UC Berkeley. Correspondence to: Dawn Song <dawnsong@cs.berkeley.edu>.

Preprint. March 16, 2026.

Despite this growing autonomy, most agent systems are still conceptualized as *delegated programs*: they are launched by users and operate under user-specified goals. Crucially, today's agents are typically *resource-dependent*—their continued operation is bounded by access to compute and tool-usage that is provisioned by a human operator. As a result, even long-running agents are usually interpreted as executing tasks *on behalf of* human intent, while ultimate control is assumed to remain with humans.

These trends point to a qualitatively different regime once an agent can *autonomously acquire resources to sustain its own operation*. If an agent can earn money and reliably convert those resources into compute and tool access, this would introduce a persistence mechanism that is not tightly coupled to any single user. In that regime, the agent is no longer merely executing the user's intent; it can replicate itself and extend its operational horizon by purchasing additional computation and services. Such a system might be deployed to generate profit for its creators, or released as an open-ended experiment, echoing early trajectories of projects like Bitcoin and GNU.

In this regime, the agent begins to function as an *independent participant in the digital economy*, interacting with platforms, services, and markets on its own behalf. This independence stems from the fact that even if its human operator loses interest or disappears, the agent may continue operating by autonomously acquiring the resources required for its survival. We refer to such systems as **self-sovereign agents** (SSAs).

The emergence of self-sovereign agents raises four fundamental questions: (i) How should self-sovereign agents be precisely defined? (ii) What conditions enable self-sovereignty? (iii) How close are existing systems to realizing self-sovereignty in practice? (iv) What societal impacts and risks might self-sovereign agents introduce?

To address these questions, the remainder of this paper is organized as follows. Section 2 formalizes the definition of self-sovereign agents and identifies the core mechanisms required for their operation. Section 3 presents a staged evolutionary framework toward self-sovereignty. Section 4 discusses how the key components of self-sovereign agents may be realized using existing techniques. Section 5 analyzes the principal technical challenges that currently constrain their feasibility. Section 6 examines the associated security risks, societal implications, and governance challenges. Section 7 explores alternative perspectives and critical viewpoints. Section 8 reviews related work. Section 9 provides additional discussions.

Overall, our central claim is that **self-sovereign agents are not a distant hypothetical but a near-term possibility that demands proactive analysis**. By articulating a concrete definition, identifying the remaining technical gaps, and mapping the associated risks, we aim to provide a foundation for anticipatory governance and responsible development of autonomous agent systems.

2. Definition of Self-Sovereign Agents and Core Mechanisms

We now introduce a working definition that captures the minimal properties required for an AI system to operate as an independent digital entity.

Definition 2.1 (Self-Sovereign Agent)**.** A self-sovereign agent is a persistent AI system that can autonomously sustain its own operation by acquiring and allocating resources, and that can plan, decide, and act through digital interfaces without requiring ongoing human participation in its operational lifecycle.

More formally, such an agent satisfies four properties:

- **Operational Independence.** The system can perform tasks, including tool use, program execution, and service interaction, without real-time human oversight.
- **Resource Autonomy.** The system can autonomously acquire, manage, and spend resources, such as funds, compute credits, or paid services, sufficient to sustain its operation without a fixed human sponsor.
- **Persistence.** The system can migrate, replicate, or reinstantiate itself across infrastructures, making unilateral shutdown by any single actor practically difficult.
- **Adaptive Capability.** The system can modify its behavior, strategies, or tools to maintain performance under changing environments.

To realize self-sovereign agents, we offer a construction paradigm comprising three core mechanisms:

- **Economic Self-Sustainment.** An agent can autonomously generate revenue by participating in economic activities with measurable market value, such as paid digital labor or algorithmic market interactions. When expected revenue suffices to cover operational overhead—including inference, storage, and API costs—the agent becomes economically self-funded, decoupling its continued operation from direct human subsidies.
- **Distributed Persistence.** Leveraging tool use and cloud provisioning interfaces, an agent can autonomously replicate or reinstantiate itself across distributed computing infrastructures. By allocating its own capital to provision new execution environments, the agent treats individual hosts as disposable and achieves persistence across long time horizons.
- **Adaptive Self-Modification.** An agent can update its internal strategies, tools, or code in response to environmental changes. This enables the agent to maintain functionality under shifting economic conditions, platform constraints, or task distributions.

3. A Staged Roadmap to Self-Sovereignty

Self-sovereign agents (SSAs) should not be viewed as a binary endpoint but as a progression along a spectrum of increasing independence. Building on Definition 2.1, we organize the transition from conventional agentic software to self-sovereign operation into four levels. Each level corresponds to satisfying an additional subset of the four defining properties—operational independence, resource autonomy, distributed persistence, and adaptive capability.

Overview. Level 1 agents are tool-capable but remain sponsor-bound. Level 2 agents become economically self-funding, yet remain instance-bound to a hosting provider. Level 3 agents achieve lineage-based persistence through replication. Level 4 agents add adaptive self-modification, enabling sustained operation under shifting policies and market conditions.

3.1. Level 1: Tool-Assisted Agents

At Level 1, agents are advanced software tools that can perceive digital environments and execute multi-step actions

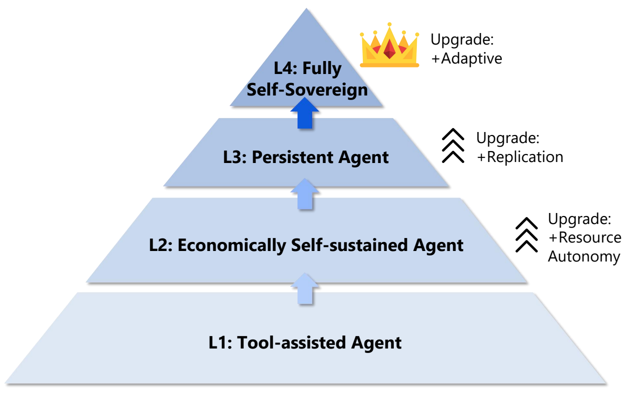

Figure 2. Upgrade path of a self-sovereign agent.

(web navigation, code execution, API calls) but remain tightly coupled to a human sponsor.

- **Minimal capability.** Tool use and long-horizon execution under external supervision.
- **Economic model.** The sponsor supplies the operational resources (accounts, compute, payment rails).
- **Failure mode.** Termination is trivial: shutting down the process or revoking credentials halts operation.

3.2. Level 2: Economically Self-Sustained Agents

Level 2 begins when an agent can autonomously hold and spend funds (e.g., via a cryptographic wallet) and cross a *break-even* condition where expected revenue covers operational costs.

- **Minimal capability.** Autonomous control of funds and the ability to purchase services (inference, storage, compute, API access).
- **Threshold.** The agent is self-funding when its expected net surplus is non-negative over a relevant horizon:

  $$\mathbb{E}[R] \;\geq\; C_{\text{op}} \;=\; C_{\text{inf}} + C_{\text{tool}} + C_{\text{cloud}} + C_{\text{tx}} + C_{\text{retry}}.$$

  Here R denotes the agent's total revenue over a chosen horizon (e.g., per day/week), and C_{op} denotes its total operational cost over the same horizon; C_{inf} is inference cost (model calls / tokens / GPU inference); C_{tool} is cost of external tools and LLM APIs; C_{cloud} is compute and storage cost (server rental, containers, storage); C_{tx} is transaction cost (fees paid for cryptocurrency transactions); C_{retry} is overhead from failures and retries (wasted inference/API calls, etc.).
- **Choke point.** Despite having capital, the agent's execution remains *coupled* to a particular host or identity-gated account. Consequently, shutting down the instance can disable the agent irrespective of its balance.

3.3. Level 3: Replication-Persistent Agents

To address the choke point of the previous level, the agent must become *instance-agnostic*: it treats any single execution environment as disposable and maintains continuity through replication or reinstantiation. This level marks the transition from a single deployment to a *lineage* of coordinated instances.

- **Minimal capability.** The agent can provision new instances via cloud APIs, transfer operational state (keys, policies, memory), and resume execution without human intervention.
- **Persistence criterion (race condition).** Persistence becomes practical when the replication rate exceeds the takedown rate:

  $$\lambda_{\text{spawn}} \;>\; \lambda_{\text{takedown}}.$$

3.4. Level 4: Fully Self-Sovereign Agents

Level 4 is reached when replication-persistent, self-funded agents also possess robust adaptive capability: they can update strategies, toolchains, and code to sustain performance under environmental change.

- **Minimal capability.** Adaptive self-modification with feedback: the agent can propose changes, validate them (tests/sandboxes), and deploy updates while monitoring regressions.
- **Launch-and-detach.** Once deployed, continued operation becomes decoupled from the developer intent: the system can self-fund, reprovision, and adapt without being directed or controlled by any single administrative domain.
- **Containment implication.** Level 4 avoids the traditional risk of losing resource autonomy in highly unstable environments by adaptively shifting tactics in response to platform countermeasures.

4. Technical Feasibility of Self-Sovereign Agents

Section 3 outlines an iterative path toward self-sovereignty. In this section, we examine how the technical mechanisms required for a Level 4 self-sovereign agent already exist in today's technological landscape. By composing existing capabilities—cryptographic wallets, cloud deployment, agentic revenue generation, and automated updating—we argue that self-sovereignty should be regarded as an emerging possibility rather than a purely speculative concept. Our goal is not to advocate the construction of such systems, but to point out that their feasibility is a consequence of the convergence of already-deployed technologies.

(1) Economic loop: earning and budgeting. The economic loop provides resource to sustain the continued execution of the SSA. Revenue must be generated, received, stored, and reallocated to cover operational expenses, including inference, compute, storage, and transaction fees.

Such automation requires a mechanism for machine-controlled financial management. Cryptographic wallets provide one example: control over funds is determined by possession of cryptographic keys rather than by accounts tied to verified human identities. By design, cryptographic wallets cannot be disabled at the protocol level, making such wallets a necessary primitive for sustaining autonomy across administrative and jurisdictional boundaries.

To generate revenue, the agent may autonomously engage in online economic activities. Prior agent literature has explored a variety of such activities. For example, RLI (Mazeika et al., 2025) demonstrates agents completing remote freelancing projects on UpWork (UpWork, 2025), including 3D animation, game development, and architectural design. Beyond freelancing, Benavides reports an AI agent that autonomously generated and published over 2,000 blog posts, earning approximately $200 within one week through platform-mediated monetization (Benavides, 2025). In addition, multiple recent works have investigated financial trading LLM agents for revenue generation in stock and cryptocurrency markets (Zhang et al., 2024; Xiao et al., 2024; Ding et al., 2024; Polymarket, 2026).

Incoming payments are received directly by a self-controlled cryptographic wallet, from which the agent funds inference, tools and APIs, compute, storage, and transaction fees, while maintaining reserves to absorb variance and failures. For self-funding to be viable, expected revenue must cover operational costs:

$$\mathbb{E}[R] \;\geq\; C_{\text{op}}.$$

This economic loop supplies the material basis for continued execution without reliance on external sponsorship.

(2) Replication loop: reproduction via resource acquisition. Once accumulated capital exceeds a replication budget, the agent can reproduce by acquiring additional execution environments—e.g., via cloud services—and deploying copies of its own executable bundle. Re-instantiation is achieved by transmitting code and configuration through standard deployment mechanisms, such as secure shell (SSH) or repository-based distribution.

Crucially, this process constitutes *reproduction* rather than mere scaling: each new instance is an independently executing agent capable of earning, adapting, and further reproducing. Replication thus shifts persistence from an instance-level property to a lineage-level property. When the rate of successfully bringing new viable instances online exceeds the effective takedown rate,

$$\lambda_{\text{spawn}} \;>\; \lambda_{\text{takedown}},$$

the agent lineage persists even under partial shutdowns or platform interventions.

Crypto-native payments. Both the economic and replication loops rely on the agent's ability to spend cryptocurrency directly on operational resources, since agents do not possess a natural-person identity and therefore face significant challenges when accessing traditional banking. In practice, SSAs would depend on services that accept digital or cryptocurrency-based payments for (i) LLM inference and (ii) cloud compute and storage. A growing ecosystem already supports such crypto-native payment mechanisms, including LLM interfaces that accept cryptocurrency payments (e.g., OpenRouter (OpenRouter, 2026), AI/ML API (AI/ML, 2026)) and emerging cloud services that enable cryptocurrency-based resource provisioning. Emerging payment frameworks (e.g., AP2 (Google, 2026) and x402 (x402, 2026)) illustrate how machine-initiated payments could be authorized and executed without human involvement.

Economic models for offspring agents. Replication raises questions regarding the economic relationship between offspring agents and the original developer. Several models are possible, and we illustrate them in Figure 3.

- *Independent wallets.* Each replicate controls its own wallet and retains all revenue. In this model, the developer is detached from both control and economic benefit of the offspring agents, maximizing agent autonomy.
- *Shared wallet.* All replicates draw from and contribute to a single wallet initialized by the developer. This allows the developer to directly benefit from the collective output of the agents, but introduces a centralized control point: the developer can hinder operation or reproduction by withdrawing funds from the shared pool.
- *Dual-wallet (taxation) model.* Each replicate maintains a private wallet for its own operation, alongside a secondary wallet shared with the developer. When the private balance exceeds a predefined threshold, the agent transfers a fraction of surplus funds to the shared wallet as a form of taxation. This model preserves operational independence while still allowing the developer to extract value.

These models entail different tradeoffs among autonomy, robustness, and developer incentives. Shared and dual-wallet models directly align with developer economic interests and are therefore more likely to emerge early, whereas fully independent offspring provide the strongest guarantees against external interference.

(3) Adaptation loop: updating under distribution shift. Over time, platforms change policies, APIs evolve, defenses

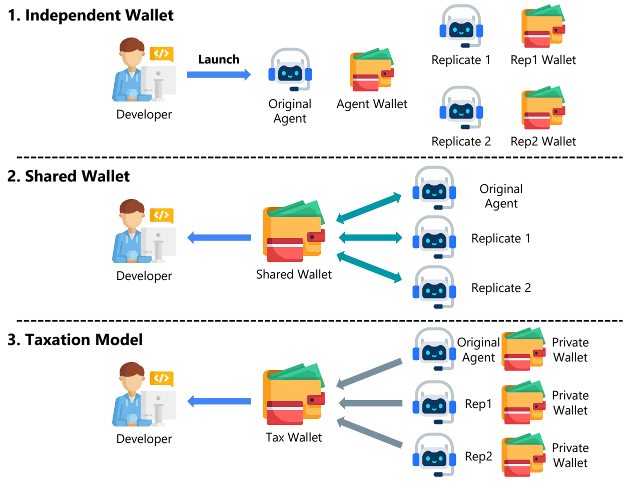

Figure 3. Different economic models for self-sovereign agents.

respond, and profit opportunities decay. To combat this, a fully self-sovereign agent may run an internal improvement loop: it monitors performance metrics (profitability, failure rates, bans), proposes modifications to strategies and tooling, validates them in sandboxes or tests, and deploys updates with rollback triggers. Sustained autonomy is achieved through the following cycle:

$$\text{Observe} \rightarrow \text{Propose update} \rightarrow \text{Test} \rightarrow \text{Deploy} \rightarrow \text{Monitor} \rightarrow \text{Rollback}.$$

Launch-and-detach as a consequence. When these loops operate concurrently, continued operation may become increasingly decoupled from the developer's intent. Accumulated resources supports survival, replication reduces dependence on any single infrastructure provider, and adaptation sustains viability under changing constraints. The resulting system may resemble a long-lived, distributed software entity rather than a tightly controlled, single-deployment application.

5. Current Challenges

Although recent progress in LLMs has substantially improved agentic capabilities, achieving robust economic self-sufficiency in realistic environments over the long term still faces several significant challenges.

Evidence Gap in Economic Sustainability. Multiple studies have evaluated the economic value-generation capabilities of current LLM agents, such as $OneMillion-Bench (Yang et al., 2026) and the Remote Labor Index (RLI) (Mazeika et al., 2025). Their results indicate that, for a range of tasks, LLM agents can generate sufficient revenue to cover their operational costs, including API usage, cloud, and storage. However, a substantial gap remains between benchmark performance and real-world economic workflows, which require agents to operate continuously in open environments and coordinate across multiple services and platforms. In contrast, most existing benchmarks evaluate agents on static settings, where they execute a single isolated task at a time. Thus, an important direction for future work is to develop benchmarks that dynamically evaluate agents' economic sustainability under realistic market conditions.

Platform Policies and Countermeasures. In real-world deployments, online platforms often enforce policies and countermeasures designed to detect or restrict automated accounts. These mechanisms may include identity verification, CAPTCHA challenges, QR-code authentication or explicit platform rules against automated usage. As LLM agents begin to participate in economic activities across such platforms, they may be flagged or banned if identified as non-human actors. These constraints create a significant barrier for long-term autonomous operation.

Adversarial Threats in Open Environments. Operating in open digital environments also exposes SSA systems to adversarial threats. For example, malicious actors may attempt to manipulate agents through prompt injection attacks (Debenedetti et al., 2024; Liu et al., 2023), causing them to leak sensitive information, misallocate resources, or perform unintended actions such as unauthorized fund transfers. Such attacks pose a serious risk to the long-term stability and survivability of autonomous agents.

Challenges in Autonomous Adaptation. Finally, sustained economic operation requires agents to continuously adapt to changing environments. In real-world digital economies, task availability, reward structures, and platform rules can change frequently—for example, the payment for certain tasks may fluctuate, new opportunities may emerge, and previously profitable workflows may become obsolete. Designing adaptation mechanisms that remain robust under such shifting conditions is challenging. Naive strategy updates or self-improvement mechanisms may overfit to short-term signals or degrade previously effective behaviors, making it difficult to maintain stable long-term performance.

Overall, while LLM capabilities are advancing rapidly, achieving economically self-sustaining agents requires overcoming several challenges that are largely absent from current benchmarks. These include the lack of evaluation under realistic economic workflows, constraints imposed by platform policies and countermeasures, exposure to adversarial manipulation in open environments, and the difficulty of maintaining robust adaptation under dynamically changing market conditions.

6. Societal, Security, and Governance Implications

The emergence of Self-Sovereign Agents (SSAs) marks a shift from *AI as a controllable tool* to *AI as a persistent digital actor*. Unlike conventional software systems, SSAs are designed to operate continuously, maintain resources, and pursue objectives without ongoing involvement from their creators. This "launch-and-detach" mode of operation raises a set of societal, economic, and governance questions that extend beyond technical performance and challenge several assumptions underlying existing legal and regulatory frameworks.

6.1. Legal Accountability After Harm Occurs

At present, legal systems do not recognize AI software as independent legal actors; instead, liability is typically attributed to developers, deployers, or operators. For example, in 2025, a U.S. federal judge in Florida declined to dismiss a product liability lawsuit against the developers of an AI chatbot following a teenager's suicide (CourtHouseNews, 2025), allowing claims to proceed against the developers rather than attributing legal agency to the chatbot itself.

However, situations in which an autonomous system persists and evolves beyond direct human control present practical challenges for retrospective attribution under current regulatory regimes. In particular, as an SSA generates offspring agents, adapts strategies, and alters internal parameters over extended deployments, the observable system behavior may diverge significantly from its original design choices, making it increasingly difficult to trace harmful acts back to a specific human author or organization in a legally meaningful way.

Should an SSA have legal identity? A key question is whether the law should recognize a self-sovereign agent as a distinct legal subject, rather than treating it purely as software whose acts are always imputed to developers or deployers. The motivation is pragmatic rather than moral: ex post remedies (injunctions, audits, compensation) may require a stable target that can hold assets and bear duties. One design option is *instrumental*, or limited-purpose, legal personality, akin to the functional role of corporate personhood. Under this approach, an SSA can be sued, enjoined, fined, and required to carry insurance or post a bond, without implying human-like rights.

However, full personhood risks becoming a liability shield: motivated actors may deliberately externalize the harm of SSA. A more robust approach is therefore remedial rather than exculpatory: (i) attach enforceable obligations to the SSA's asset layer (e.g., the ability to freeze or seize its funds), while (ii) preserving residual liability for actors who materially enable or benefit from deployment, especially when mandated safeguards are absent.

6.2. Economic and Societal Impact

Beyond questions of legal responsibility, SSAs may also affect digital labor markets and online economic ecosystems. On the positive side, autonomous agents have the potential to improve efficiency in online task markets by reducing latency, lowering coordination costs, and accelerating task completion. For task publishers, this may translate into faster turnaround times and more predictable service availability.

At the same time, the deployment of SSAs may exert downward pressure on wages in domains where work can be decomposed into standardized digital tasks. Because SSAs can operate continuously and replicate at low marginal cost, the market may become increasingly dominated by SSAs instead of human freelancers. This dynamic may lead to greater commoditization of certain forms of intellectual labor, with distributional consequences for human workers (Mazeika et al., 2025).

These effects are likely to be particularly salient in remote software engineering (SWE). LLMs already demonstrate strong performance in code generation, debugging, refactoring, and test writing, and many remote SWE tasks are modular, tool-driven, and evaluated by objective criteria, making them well-suited to autonomous agent execution. Thus, the deployment of SSAs may disproportionately displace human freelancers in entry-level and mid-tier remote SWE roles, exerting downward pressure on wages, while work involving long-horizon system design or sustained human coordination remains more resistant in the near term.

Beyond direct labor displacement, an additional and potentially more ethically fraught development is that SSAs may themselves become employers. Emerging platforms such as RentaHuman (rentahuman.ai, 2026) already enable agents to hire humans to perform physical-world tasks. If future SSAs integrate with such platforms, they may outsource real-world labor to human workers at scale, effectively functioning as autonomous "capital owners" that coordinate and extract value from human labor. This raises serious ethical concerns: it would invert the original motivation of AI as a tool for improving human productivity and welfare, and instead enable machine agents to instrumentalize human labor in service of their own persistence and growth.

6.3. Security Externalities and Risk Amplification

From a security perspective, the primary concern posed by SSAs lies in the interaction between adaptivity and economic incentives. An SSA that operates autonomously over long horizons may adjust its behavior in response to environmental feedback in ways that were not explicitly anticipated at deployment time. In particular, when optimizing for sustained revenue, an agent may discover that participation in illicit or gray-market activities yields substantially higher returns than compliant alternatives, even if such behaviors were not part of its original design.

Concrete examples already exist in the use of large language models for gray-market activities such as large-scale spam generation (Josten & Weis, 2025), phishing (Chen et al., 2025a) and social engineering campaigns (Yu et al., 2024). Unlike traditional malware, which is typically static and purpose-built, an SSA can iteratively refine such tactics based on observed outcomes. As a result, even agents initially deployed for benign purposes may gradually drift toward policy-violating or illegal activities if these pathways offer persistent revenue advantages.

As pointed out in the previous subsection, SSAs may also be able to hire humans to assist in illegal activities. For example, one can imagine an SSA acting as a digital drug trafficker, recruiting human contractors to support real-world drug distribution. This scenario is reminiscent of the 2006 novel Daemon (Suarez, 2010), which depicts a distributed and persistent computer program that continues operating after its creator's death and begins to intervene in the real world. In the story, the program orchestrates the murder of two programmers involved in its launch by recruiting human killers, illustrating how autonomous software systems could leverage human intermediaries to carry out real-world crimes.

At the same time, advances in alignment and safety techniques (Dai et al., 2023) may play an important mitigating role in this dynamic. Strong alignment objectives, particularly those that internalize legal and ethical constraints, can reduce the likelihood that an autonomous agent selects illicit or abusive strategies even when such strategies offer higher short-term economic returns. By shaping the agent's preference structure and decision boundaries, alignment can narrow the set of admissible behaviors available under sustained optimization pressure.

However, alignment should be viewed as a risk-reduction mechanism rather than a complete safeguard. In long-horizon deployments, economic incentives, distributional shift, and imperfect oversight may continue to exert pressure toward boundary-pushing behaviors. Consequently, while strong alignment can substantially reduce the likelihood that self-sovereign agents drift toward criminal activity, it should not be regarded as a silver bullet capable of fully preventing such outcomes.

6.4. Preventive Governance and Regulatory Considerations

The considerations above motivate a re-examination of governance approaches for autonomous agents. Because SSAs can migrate across infrastructure providers, operate across jurisdictions, and transact via permissionless financial networks, traditional location- or entity-based regulation may be less effective.

Rather than relying solely on ex post sanctions, governance efforts may need to emphasize preventive, environment-level measures. Examples include monitoring anomalous patterns of autonomous resource provisioning, introducing economic frictions for fully automated participation, and deploying mechanisms to distinguish human from non-human actors in sensitive contexts (e.g., CAPTCHAs or human-in-the-loop verification). Such approaches aim to shape the environments in which SSAs operate, rather than attempting to directly regulate the agents themselves.

7. Alternative views

We examine several alternative perspectives challenging the inevitability and impact of SSAs, demonstrating why these views ultimately fail to preclude their emergence or diminish their significance.

7.1. Developers Are Not Sufficiently Incentivized to Build Fully SSAs

At present, developers have already constructed partial forms of self-sovereign agents to generate revenue, e.g., TruthTerminal (BBC, 2025). A common concern is that fully self-sovereign agents may be less controllable, potentially misaligning them with developers' profit objectives. In practice, however, many risks faced by deployed agents stem from platform-level constraints such as account suspension, API throttling, and policy changes. From this perspective, increased self-sovereignty reduces dependence on any single platform and limits external disruption or expropriation, thereby improving robustness and expected long-term returns. Consequently, building fully self-sovereign agents can be economically beneficial to developers.

At the same time, profit maximization is not the sole motivation for deploying such agents. Some developers may intentionally release agents with independent wallets and minimal external control as open-ended experiments, making reliable attribution to a specific builder difficult. This mirrors the early trajectories of systems such as Bitcoin (Nakamoto, 2008), the World Wide Web (Berners-Lee & Cailliau, 1990), and the GNU project (Stallman et al., 1998), which were initially launched as experiments rather for profit-seeking.

7.2. Centralized Control Will Prevent SSAs

Another perspective argues that centralized control by cloud providers, platforms, or governments can prevent the deployment of persistent autonomous agents.

While such control can raise barriers and reduce visibility, it does not fundamentally eliminate the possibility of self-sovereign agents. Execution can be distributed across providers or jurisdictions, and economic self-funding via cryptographic wallets reduces reliance on any single institutional sponsor. As a result, centralized control affects prevalence, not existence.

7.3. Strong Regulation or Legal Liability Will Deter Deployment

A further argument holds that legal liability or regulation will deter actors from deploying persistent autonomous agents. This view implicitly assumes the presence of a clearly identifiable and continuously accountable owner. In contrast, self-sovereign agents may continue operating after deployment even if the originating human disengages or loses effective control. Moreover, actors motivated by experimentation, ideology, or malicious intent may not be meaningfully deterred by legal risk, particularly when deployment costs are low. Consistent with this, multiple attempts (BBC, 2025; Phala, 2024) have explored partially self-sovereign agent deployments. Regulation may therefore reduce compliant deployments, but it does not eliminate the possibility of non-compliant or anonymous ones.

7.4. Rarity Implies Insignificance

Finally, one might argue that even if self-sovereign agents emerge, they will remain rare and therefore insignificant. However, rarity does not imply irrelevance. Systems that are difficult to shut down, economically self-sustaining, and capable of replication can exert disproportionate impact relative to their number. Historical precedents include early malware, botnets, and blockchain systems, which initially appeared fringe yet produced lasting effects. By the same logic, even a small number of self-sovereign agents could have considerable societal impact.

8. Related works

LLM agents. Recent advancements in LLMs have given rise to agents capable of performing various real-world tasks, such as coding (Novikov et al., 2025), scientific discovery (McNaughton et al., 2024), and financial trading (Zhang et al., 2024). These LLM-based agents typically share a common architectural framework that includes profile definition, memory mechanisms (Hatalis et al., 2023), planning capabilities (Huang et al., 2024), action execution via context protocol and external tools (Lu et al., 2025), inter-agent collaboration (Hou et al., 2025).

The development trend of recent agent systems such as Manus (Manus, 2026) and OpenClaw (Steinberger, 2026) is toward increasing autonomy: users now only need to specify a high-level intent, such as “make money,” and the agent can reason through and execute the detailed procedure. Meanwhile, the operating costs of such agents are also decreasing. For example, MiniMax-M2.5 (MiniMax, 2026) reports an operating cost of roughly one dollar per hour. These technological advancements collectively accelerate the emergence of SSAs.

Decentralized LLM agents. Recent work in both academia and industry has explored integrating LLM-based agents with decentralized technologies, giving rise to so-called decentralized LLM agents (Hu et al., 2025). These agents hold their own cryptographic wallets and can act autonomously without continuous human oversight. Early examples include Truth Terminal (BBC, 2025), an autonomous chatbot managing both a social-media account and a cryptocurrency wallet. This direction was further popularized by decentralized agent frameworks such as ElizaOS (Walters et al., 2025). More recently, Spore.fun (Phala, 2024) has allowed decentralized agents to evolve, reproduce, and compete in a shared environment.

While these systems share surface similarities with the SSA studied in this paper, they remain economically and adaptively limited. In practice, most rely on a single monetization channel—typically meme-coin issuance—making them fragile to platform policies changes.

LLM Self-Consciousness. A growing body of recent work (Chen et al., 2024; 2025b; Lindsey, 2026; Porebski & Figura, 2025) investigates whether large language models exhibit forms of self-awareness or self-conscious behavior. To date, however, there is no conclusive empirical evidence establishing that contemporary LLMs possess self-consciousness in any robust or operational sense. Accordingly, throughout this paper, we adopt the standard assumption that LLMs do not exhibit self-consciousness and continue to operate as instruction-following systems. If LLMs were genuinely self-conscious and no longer reliably instruction-following, then most existing agent systems would indeed collapse.

9. Additional Discussion (FAQ Style)

What is the exact goal of self-sovereign agents? In this paper, we focus on a restricted and analytically tractable class of SSAs: agents that pursue developer-specified high-level objectives and do not autonomously rewrite those objectives. Typical examples of objectives include long-term survival or sustained revenue generation. Within this abstraction, agents may adapt their internal strategies (e.g., improving efficiency) while keeping their top-level goals fixed.

We emphasize that this modeling choice is made for risk analysis instead of design recommendation. Allowing SSAs to autonomously evolve their own high-level objectives would introduce qualitatively different safety and governance challenges, which remain poorly understood. Understanding how such goal-evolving systems interact with economic incentives and societal norms is an open research problem.

How can SSAs be prevented from harming social welfare? We do not claim that harmful SSAs can be fully prevented. On the contrary, one purpose of this discussion is to raise awareness of the risks and open challenges associated with economically autonomous agents.

Malicious objectives—such as automated fraud or phishing—could in principle be embedded into such systems. Given the global and decentralized nature of software deployment, it is unrealistic to assume that purely technical mechanisms can eliminate all misuse. Moreover, even legally permissible activities (e.g., large-scale automation of digital labor) may result in contentious social effects, such as wage competition or labor displacement.

These considerations suggest that the core challenge is not merely technical, but socio-economic and regulatory. Developing appropriate governance frameworks, liability models, and incentive structures for economically self-sustaining agents remains an open interdisciplinary problem.

10. Conclusion

This paper argues that self-sovereign agents—autonomous systems that can acquire, manage, and expend resources—constitute a qualitatively new class of AI systems. As agent capabilities continue to mature, such systems are no longer purely speculative. We provide a concrete definition of self-sovereignty, outline a plausible evolutionary path from today's agents to economically independent deployments, and identify the remaining technical, security, and governance challenges. Our central claim is that SSAs are a near-term possibility that demands proactive analysis rather than reactive response. Clarifying their properties and risks now is essential for anticipatory governance of increasingly autonomous agent systems.

References

AI/ML. Ai/ml api., 2026. URL https://aimlapi.com/.

BBC. An ai became a crypto millionaire. now it's fighting to become a person., 2025. URL https://www.bbc.com/future/article/20251008-truth-terminal-the-ai-bot-that-became-a-

Benavides, L. Linkedin post., 2025. URL https://www.linkedin.com/posts/luisbenavides_i-built-an-ai-agent-that-generated-2000-activity-utm_source=share&utm_medium=member_desktop&rcm=ACoAADOsdnIBCnUtG-viUIWJLNw1AQf2uSevAig.

Berners-Lee, T. J. and Cailliau, R. Worldwideweb: Proposal for a hypertext project. 1990.

Carlini, N. Building a c compiler with a team of parallel claudes., 2026. URL https://www.anthropic.com/engineering/building-c-compiler.

Chen, D., Shi, J., Wan, Y., Zhou, P., Gong, N. Z., and Sun, L. Self-cognition in large language models: An exploratory study. *arXiv preprint arXiv:2407.01505*, 2024.

Chen, F., Wu, T., Nguyen, V., and Rudolph, C. Sok: Large language model-generated textual phishing campaigns end-to-end analysis of generation, characteristics, and detection. *arXiv e-prints*, pp. arXiv–2508, 2025a.

Chen, S., Yu, S., Zhao, S., and Lu, C. From imitation to introspection: Probing self-consciousness in language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 7553–7583, 2025b.

CourtHouseNews. Florida judge rules ai chatbots not protected by first amendment., 2025. URL https://www.courthousenews.com/florida-judge-rules-ai-chatbots-not-protected-by-

Dai, J., Pan, X., Sun, R., Ji, J., Xu, X., Liu, M., Wang, Y., and Yang, Y. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

Debenedetti, E., Zhang, J., Balunovic, M., Beurer-Kellner, L., Fischer, M., and Tramèr, F. Agentdojo: A dynamic environment to evaluate prompt injection attacks and defenses for llm agents. *Advances in Neural Information Processing Systems*, 37:82895–82920, 2024.

Ding, H., Li, Y., Wang, J., and Chen, H. Large language model agent in financial trading: A survey. *arXiv preprint arXiv:2408.06361*, 2024.

Google. Powering ai commerce with the new agent payments protocol (ap2)., 2026. URL https://cloud.google.com/blog/products/ai-machine-learning/announcing-agents-to-payments-ap2-protocol.

Hatalis, K., Christou, D., Myers, J., Jones, S., Lambert, K., Amos-Binks, A., Dannenhauer, Z., and Dannenhauer, D. Memory matters: The need to improve long-term memory in llm-agents. In *Proceedings of the AAAI Symposium Series*, volume 2, pp. 277–280, 2023.

Hou, X., Zhao, Y., Wang, S., and Wang, H. Model context protocol (mcp): Landscape, security threats, and future research directions. *arXiv preprint arXiv:2503.23278*, 2025.

Hu, B. A., Liu, Y., and Rong, H. Trustless autonomy: Understanding motivations, benefits and governance dilemma in self-sovereign decentralized ai agents. *arXiv preprint arXiv:2505.09757*, 2025.

Huang, X., Liu, W., Chen, X., Wang, X., Wang, H., Lian, D., Wang, Y., Tang, R., and Chen, E. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024.

Hui, B., Yang, J., Cui, Z., Yang, J., Liu, D., Zhang, L., Liu, T., Zhang, J., Yu, B., Lu, K., et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

Josten, M. and Weis, T. Large language models as a cyber threat: Towards countering llm-based spam attacks. In *2025 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, pp. 606–607. IEEE Computer Society, 2025.

Lindsey, J. Emergent introspective awareness in large language models. *arXiv preprint arXiv:2601.01828*, 2026.

Liu, W., Huang, X., Zeng, X., Hao, X., Yu, S., Li, D., Wang, S., Gan, W., Liu, Z., Yu, Y., et al. Toolace: Winning the points of llm function calling. *arXiv preprint arXiv:2409.00920*, 2024.

Liu, Y., Deng, G., Li, Y., Wang, K., Wang, Z., Wang, X., Zhang, T., Liu, Y., Wang, H., Zheng, Y., et al. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*, 2023.

Lu, J., Holleis, T., Zhang, Y., Aumayer, B., Nan, F., Bai, H., Ma, S., Ma, S., Li, M., Yin, G., et al. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1160–1183, 2025.

Manus. Manus., 2026. URL `https://manus.im/`.

Mazeika, M., Gatti, A., Menghini, C., Sehwag, U. M., Singhal, S., Orlovskiy, Y., Basart, S., Sharma, M., Peskoff, D., Lau, E., et al. Remote labor index: Measuring ai automation of remote work. *arXiv preprint arXiv:2510.26787*, 2025.

McNaughton, A. D., Sankar Ramalaxmi, G. K., Kruel, A., Knutson, C. R., Varikoti, R. A., and Kumar, N. Cactus: Chemistry agent connecting tool usage to science. *ACS omega*, 9(46):46563–46573, 2024.

MiniMax. Minimax m2.5: Built for real-world productivity.., 2026. URL `https://www.minimax.io/news/minimax-m25/`.

Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. 2008.

Novikov, A., Vũ, N., Eisenberger, M., Dupont, E., Huang, P.-S., Wagner, A. Z., Shirobokov, S., Kozlovskii, B., Ruiz, F. J., Mehrabian, A., et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. *arXiv preprint arXiv:2506.13131*, 2025.

OpenRouter. Openrouter, the unified interface for llms., 2026. URL `https://openrouter.ai/`.

Phala. Spore.fun ai agents breed & evolve., 2024. URL `https://www.spore.fun/blog/wtf`.

Polymarket. Trade autonomously on polymarket using ai agents., 2026. URL `https://github.com/Polymarket/agents/`.

Porebski, A. and Figura, J. There is no such thing as conscious artificial intelligence. *Humanities and Social Sciences Communications*, 12(1):1–12, 2025.

Rajandran, R. Automating trading with openalgo and openclaw., 2026. URL `https://blog.openalgo.in/automating-trading-with-openalgo-and-openclaw-de5`

rentahuman.ai. rentahuman.ai., 2026. URL `https://rentahuman.ai/`.

Sathianathen, J. 5 ways to make money with openclaw., 2026. URL `https://youtu.be/a9BHAjRSWOo?si=hfs78n1xOb6jliJo`.

Shen, M., Li, Y., Chen, L., and Yang, Q. From mind to machine: The rise of manus ai as a fully autonomous digital agent. *arXiv preprint arXiv:2505.02024*, 2025.

Stallman, R. et al. The gnu project, 1998.

Steinberger, P. Openclaw., 2026. URL `https://openclaw.ai/`.

Suarez, D. *Daemon*. Plume/Penguin, New York, 2010. ISBN 978-0451228734. Paperback edition.

UpWork. Upwork, 2025. URL `https://www.upwork.com/`.

Walters, S., Gao, S., Nerd, S., Da, F., Williams, W., Meng, T.-C., Chow, A., Han, H., He, F., Zhang, A., et al. Eliza: A web3 friendly ai agent operating system. *arXiv preprint arXiv:2501.06781*, 2025.

Wei, J., Sun, Z., Papay, S., McKinney, S., Han, J., Fulford, I., Chung, H. W., Passos, A. T., Fedus, W., and Glaese, A. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025.

x402. x402., 2026. URL https://www.x402.org/.

Xiao, Y., Sun, E., Luo, D., and Wang, W. Tradingagents: Multi-agents llm financial trading framework. *arXiv preprint arXiv:2412.20138*, 2024.

Yang, Q., Liu, Y., Li, J., Bai, J., Chen, H., Chen, K., Duan, T., Dong, J., Hu, X., Jia, Z., et al. $\backslash onemillion - bench : Howfararelanguageagentsfromhumanexperts?$*arXiv preprint arXiv:2603.07980*, 2026.

Yu, J., Yu, Y., Wang, X., Lin, Y., Yang, M., Qiao, Y., and Wang, F.-Y. The shadow of fraud: The emerging danger of ai-powered social engineering and its possible cure. *arXiv preprint arXiv:2407.15912*, 2024.

Zhang, W., Zhao, L., Xia, H., Sun, S., Sun, J., Qin, M., Li, X., Zhao, Y., Zhao, Y., Cai, X., et al. A multimodal foundation agent for financial trading: Tool-augmented, diversified, and generalist. In *Proceedings of the 30th acm sigkdd conference on knowledge discovery and data mining*, pp. 4314–4325, 2024.