When Code Becomes an "Ephemeral" Tool: Governance Crisis in the Age of AI Agents and the Response of Logographic AI

Abstract

In June 2026, Zhenfeng Cao published a paper titled Agentic Software: How AI Agents Are Restructuring the Software Paradigm, arguing that AI agents are transforming software from a "static normative carrier" into "dynamically generated ephemeral tools," marking a fundamental paradigm shift from SaaS to Agent-as-a-Service (AaaS)[1]. Cao's paper focuses on technical paradigm analysis and does not directly address sovereignty or auditing issues. A social media blogger, on September 1, 2026, extended this technical judgment into a warning for public governance — that the "end of static code" would lead to a "crisis of public auditability"[2]. This paper takes Cao's paper and its derivative discussions as a point of departure, introduces the Logographic AI framework as a response, and argues that only by restructuring the auditability of reasoning at the cognitive primitive level can the accountability foundation of public governance be rebuilt under the AaaS paradigm.

Keywords: Agentic Engineering; AaaS; Public Auditability; Logographic AI; Morpho-Root Hard Constraints; Auditable Reasoning Chains

I. The Core Thesis of Agentic Engineering

In June 2026, Zhenfeng Cao, in his paper Agentic Software: How AI Agents Are Restructuring the Software Paradigm, put forward a disruptive thesis[1]: AI agents are transforming software from a "static normative carrier" into "dynamically generated ephemeral tools."

The paper's core contribution lies in formally distinguishing two paradigms:

Dimension	**Traditional Software Engineering**	**Agentic Engineering**
Core Object of Study	Static source code	Dynamic agentic systems
Decision Logic	Predetermined at coding time	Dynamically generated at runtime
Control Model	Human-predefined	LLM-driven reasoning loop
Human Role	Code writer	Intent architect

The paper argues that traditional software systems are constrained by a fixed upper limit of human cognitive capacity—when problem complexity exceeds human capability, no amount of traditional engineering can solve it. Agentic systems, through LLM-driven reasoning loops, decouple solution generation capacity from human cognitive limits. "This is not a 10% improvement; it's a qualitative shift in what kinds of economic problems can be solved."[1]

The paper outlines three historical stages of software delivery[1]:

Stage	Mode	Execution Location	Complexity Bearer
Software 1.0	Licensed Software	Local	End users
Software 2.0	SaaS	Cloud	Vendors
Software 3.0	AaaS	Agent-autonomous	Agents themselves

In the Agent-as-a-Service (AaaS) paradigm, code becomes an ephemeral tool serving the LLM reasoning loop—with each interaction, the system autonomously generates the code needed to complete the current task, executes it, and then discards it. The next interaction generates anew.

II. The Derivative Discussion on Governance Sovereignty

Cao's paper does not directly address governance sovereignty or government auditing issues. The sovereignty-level discussion comes from a social media blogger's extended interpretation of the paper. The blogger's post expands the paper's content into several key points[2], including:

• The End of Static Code: Code no longer exists in any repository; decision logic is no longer fixed in a reviewable carrier.

• The Crisis of Public Auditability: Administrative law requires traceability. But if the code that executed a decision was ephemeral and no longer exists in any repository, how can a court of auditors or a judge review a state decision?

• Cognitive Capture: States without autonomous reasoning infrastructure will see their public policies subjected to the biases, restrictions, and ethical alignments imposed by private LLM providers.

These extended interpretations point to a genuine governance dilemma—when decision logic is no longer fixed in reviewable code, the traceability foundation of law-based administration is eroding. Although the paper's technical analysis does not directly respond to this dilemma, the questions raised by the blogger are of serious discursive value.

III. From "Output Risk" to "Action Risk"

The paper's core thesis—"agent as software"—implies that the center of gravity of risk is shifting from "what the model outputs" to "what action permissions the system obtains."

The paper describes software evolution as a process of "complexity transferring from end users to agents"[1]. When agents assume the dual role of decision-maker and executor, the object of auditing shifts from "code" to "agent behavior"—decision logic no longer resides in any static repository but exists only in the instant of reasoning. This presents a fundamental challenge to the traditional "pre-deployment audit" model.

The blogger's "crisis of public auditability" is valid precisely because of the runtime decision-logic generation characteristic described in the paper. The paper states that "in agentic systems, decision logic is generated at runtime"[1]—a fundamental difference from traditional software where "decision logic is fixed at coding time." When this characteristic is applied to public governance scenarios, the "audit untraceability" that the blogger fears becomes a structural predicament.

When the AaaS paradigm makes "decision logic generated and disappearing at runtime" the norm,

the governance response cannot remain at the level of external requirements like "requiring AI companies to keep logs." Rather, it must restructure the generative mechanism of auditability at the cognitive primitive level. This is precisely the core question that the Logographic AI framework addresses.

IV. Logographic AI's Response: Auditable Reasoning Chains

The crisis that the blogger extends lies at root in the separation of reasoning and execution—the reasoning process is unauditable, and the execution code is ephemeral. This is precisely the core problem that Logographic AI theory seeks to address.

Logographic AI is a new AI paradigm proposed by Liu Shen in 2025[3][4]. Its core proposition is to replace the "unanchored Token" with the "Morpho-Root"—the smallest meaningful unit of Chinese characters—as the cognitive primitive of AI. Each Morpho-Root is a structured unit of meaning containing hard constraint markers such as "unbreakable." Reasoning paths that violate hard constraints are logically blocked at the generation stage—not "statistically improbable," but "logically unreachable." This mechanism makes safety a constitutive attribute of the cognitive primitive, rather than an externally attached rule.

The rigorous mathematical definitions and formal system of Morpho-Root theory—including the triple definition of Morpho-Root, the Morpho-Root semigroup and category structures, and the hard constraint propagation and blocking completeness theorems—have been established elsewhere[5], with accompanying open-source code implementations[6]. The engineering paper From Morpho-Root Theory to Engineering Practice: OpenMorpho — Implementation and Verification of an End-to-End Inference Pipeline for Logographic AI[7] and its accompanying code[8] have verified the feasibility of the end-to-end inference pipeline.

More importantly, every step of reasoning in the Logographic AI architecture is recorded as a structured **auditable reasoning chain**—including fields such as path ID, activated Morpho-Root sequence, triggered relation functions, hard constraint verification results, and blocking reasons. Even if the code disappears after execution, the reasoning chain itself remains as structured data—available for auditing, traceability, and accountability[7].

Logographic AI's response is a structural complement to the AaaS paradigm: not using "better logs" to solve "more complex black boxes," but ensuring from the cognitive primitive level that the reasoning process itself is inherently auditable, traceable, and accountable.

V. Conclusion

Zhenfeng Cao's paper reveals a fundamental transformation in the software paradigm: code has shifted from the "embodiment of legal logic" to the "scratch paper of LLM reasoning"[1]. A social media blogger extended this technical judgment into a warning for public governance—after static code disappears, how do we review decisions?[2]

Logographic AI's "Morpho-Root Hard Constraints" and "Auditable Reasoning Chains" framework[3][4][5][7] is the only technical path capable of rebuilding "traceability" under the AaaS paradigm. Not by backing up code, but by transforming the cognitive primitive itself—endowing the reasoning chain with inherent auditability at the moment of generation.

When the ephemerality of code threatens the traceability of law, the answer is not to try to freeze code, but to leave the reasoning itself with indelible, structured traces.

References

[1] Cao Z. Agentic Software: How AI Agents Are Restructuring the Software Paradigm[J/OL]. arXiv, 2026. arXiv:2606.05608. DOI:10.48550/arXiv.2606.05608.

[2] Ramírez A (@adrianramirez). TERREMOTO SOFTWARE GUBERNAMENTAL[EB/OL]. X (formerly Twitter), (2026-09-01). https://x.com/_adrianramirez_/status/2094492308548747587?s=46

[3] 刘深. 表意 AI vs. 表音 AI：AI 新范式与去殖民化宣言[J/OL]. ChinaXiv, 2025. DOI:10.12074/202503.00051.

[4] Liu S. The Rooted Intelligence: The Paradigm Revolution of Logographic AI[M]. Zenodo, 2026. DOI:10.5281/zenodo.22220783

[5] Liu S. The Formal Foundations of Morpho-Root Theory: Mathematical Definitions and Reasoning Systems for Logographic AI Cognitive Primitives[J/OL]. ChinaXiv, 2026. T202608.00490.
https://chinaxiv.org/businessFile/T202608/T202608.00490v1/T202608.00490v1.pdf

[6] WindSeaAI. morpho-root-formalization: Mathematical Definitions and Reasoning Systems for Logographic AI Cognitive Primitives[CP/OL]. Zenodo, 2026. DOI:10.5281/zenodo.22177031

[7] Liu S. From Morpho-Root Theory to Engineering Practice: OpenMorpho — Implementation and Verification of an End-to-End Inference Pipeline for Logographic AI[J/OL]. ChinaXiv, 2026. T202608.00048.
https://chinaxiv.org/businessFile/T202608/T202608.00048v1/T202608.00048v1.pdf

[8] WindSeaAI. OpenMorpho: End-to-End Inference Pipeline for Logographic AI[CP/OL]. Zenodo, 2026. DOI:10.5281/zenodo.22204882