

WHITE PAPER

The Enterprise Software Control Plane

Operationalizing Software
Governance at AI Speed



CLAIRVOYANT

Contents

P.3 Introduction

The expanding enterprise software ecosystem, the challenges of scaling cross-functional governance, and an introduction to the Enterprise Software Control Plane.

P.4 Why Software Governance is Changing

How AI, rapid development, and complex dependencies drive the need for continuous, evidence-based software evaluation.

P.5 Why It Matters

Supply chain incidents reveal the limits of traditional trust and the growing need for independent behavioral evidence.

P.6 From Risk Signals to Software Governance Decisions

Explores the gap between known security attributes and the behavioral evidence needed to govern rapidly changing software.

P.8 Introducing the Enterprise Software Control Plane

Introduces the Enterprise Software Control Plane as an independent layer to automate evidence-based governance before software deployment.

P.10 What the Enterprise Software Control Plane Does

Details the four core functions for observing software, understanding behavior, evaluating enterprise requirements, and supporting software decisions.

P.11 Automating Governance Across Risk Frameworks

Mapping single behavioral assessments to multiple security, AI, and compliance objectives via Clairvoyant Risk Frameworks.

P.15 Conclusion

Summarizes how the Enterprise Software Control Plane provides a scalable, independent foundation for consistent governance across modern ecosystems.



Introduction

Software enters enterprise environments through more channels than ever before. Applications are developed internally, acquired from third-party suppliers, assembled from open-source components, delivered through SaaS platforms, generated with AI assistance, and increasingly created through vibe coding and citizen development platforms. Software also continues to evolve after deployment through updates, integrations, and automated workflows.

As software sources expand, organizations must make a growing number of software governance decisions within enterprise environments.

These decisions extend beyond security teams. Software governance affects procurement, compliance, privacy, operational resilience, AI governance, software supply chain programs, and enterprise risk management.

Before software is deployed, executed, distributed, or granted access to enterprise systems and data, organizations must determine whether that software aligns with their requirements and risk tolerance.

The key challenge is operationalizing enterprise software governance at scale across an increasingly diverse software ecosystem.

This paper builds upon Clairvoyant's previous research examining the changing nature of software trust and the compression of software governance timelines. It extends those concepts by exploring how organizations can operationalize software governance through an Enterprise Software Control Plane.



Why Software Governance Is Changing

The software landscape is changing rapidly. Modern software increasingly incorporates AI-generated code, autonomous development tools, continuously evolving third-party components, and complex open-source dependencies. Frontier AI is also beginning to participate more directly in software development and improvement, further increasing the speed and scale of software change.

These developments are changing how organizations govern software. Organizations can no longer rely solely on supplier reputation, vulnerability status, or other known software attributes. Independent evidence of software behavior and operational impact is increasingly required before software is approved, deployed, or continuously operated within enterprise environments.

The following examples illustrate how software governance challenges are emerging across different stages of today's software lifecycle.

At the same time, software production itself continues accelerating. AI-generated code enters development pipelines faster. Open-source ecosystems evolve continuously.



Why It Matters

Recent software supply chain incidents demonstrate that software risk is no longer confined to a single stage of development or distribution. Modern attacks increasingly target AI-assisted development, package selection, trusted software frameworks, signed applications, and legitimate software distribution channels.

Collectively, these incidents illustrate how traditional trust indicators can be bypassed across multiple stages of the software lifecycle.

Incident	Stage of the Software Lifecycle	Industry Observation
Notepad++ Supply Chain Compromise	Software updates and distribution	Legitimate software update infrastructure can be abused to distribute malicious code.
Microsoft Mastra Compromise	Trusted software frameworks	Legitimate software frameworks can become software supply chain entry points.
MacSync Malware Campaign	Signed software distribution	Digital signatures and notarization do not guarantee safe software behavior.
LiteLLM Poisoning	AI software components	AI software components and dependencies can become software supply chain attack vectors.
OpenClaw Supply Chain Incident	Open-source software ecosystem	Repository and ecosystem reputation do not guarantee expected software behavior.

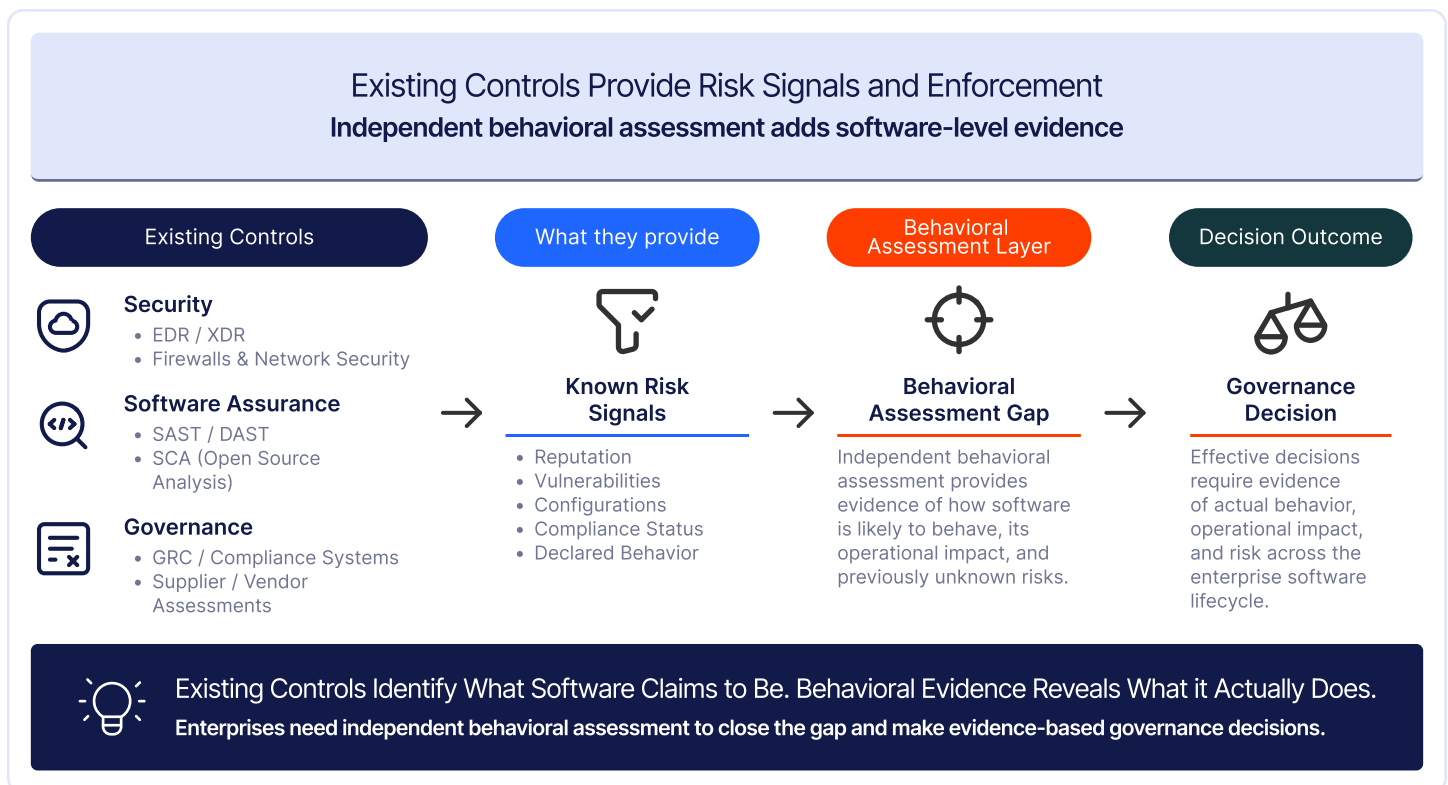
These incidents span multiple software ecosystems while exposing a common pattern. Software can no longer be evaluated solely by its source, supplier, repository, digital signature, or other traditional trust indicators.

Organizations increasingly require independent evidence of software behavior, operational impact, dependencies, and organizational risk to support decisions before deployment and as software changes over time.

From Risk Signals to Software Governance Decisions

Organizations already maintain substantial investments in security, software assurance, governance, procurement, and compliance programs. These capabilities provide valuable information regarding software vulnerabilities, suppliers, provenance, compliance requirements, and other known software attributes.

As software sources continue to expand through third-party suppliers, open-source ecosystems, AI-assisted development, citizen development, software updates, and software marketplaces, organizations are increasingly required to make software governance decisions across a broader and more dynamic software ecosystem.



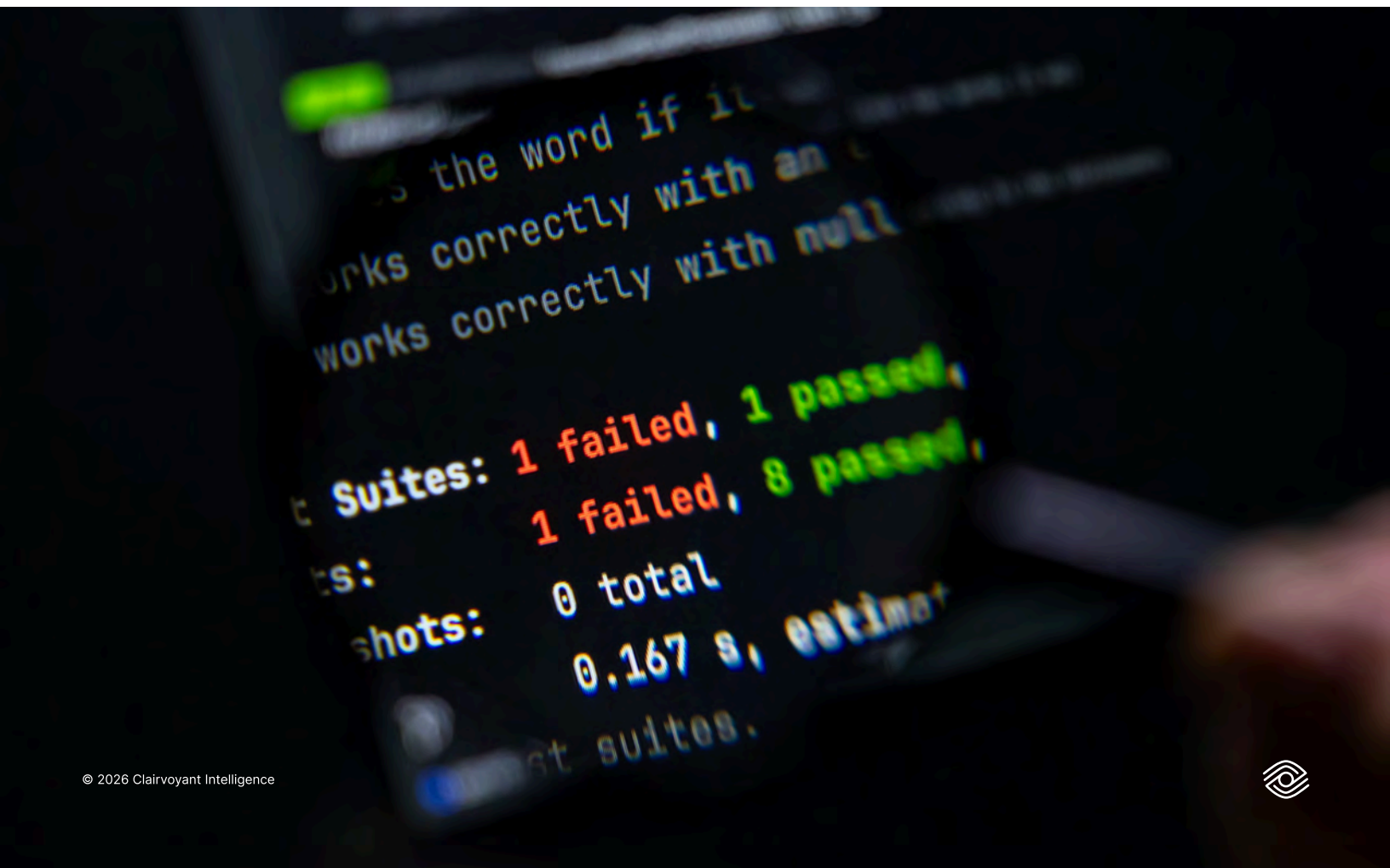
However, modern software supply chain incidents increasingly demonstrate that software behavior can change in ways that are not fully explained through known attributes alone. Legitimate software packages, trusted suppliers, approved repositories, and software updates may introduce new dependencies, permissions, communications patterns, operational behaviors, or other functionality that was not anticipated during traditional review processes.

The challenge grows as frontier AI takes a larger role in software development, vulnerability discovery, automated fix generation, and software delivery workflows.

Software can now be created, modified, and updated at a pace that increasingly compresses traditional review, approval, and governance timelines.

Organizations require sufficient evidence regarding software behavior and operational impact before software is deployed, distributed, executed, or granted access to enterprise systems.

A gap remains between the known attributes captured by existing controls and the behavioral evidence required to support software decisions.

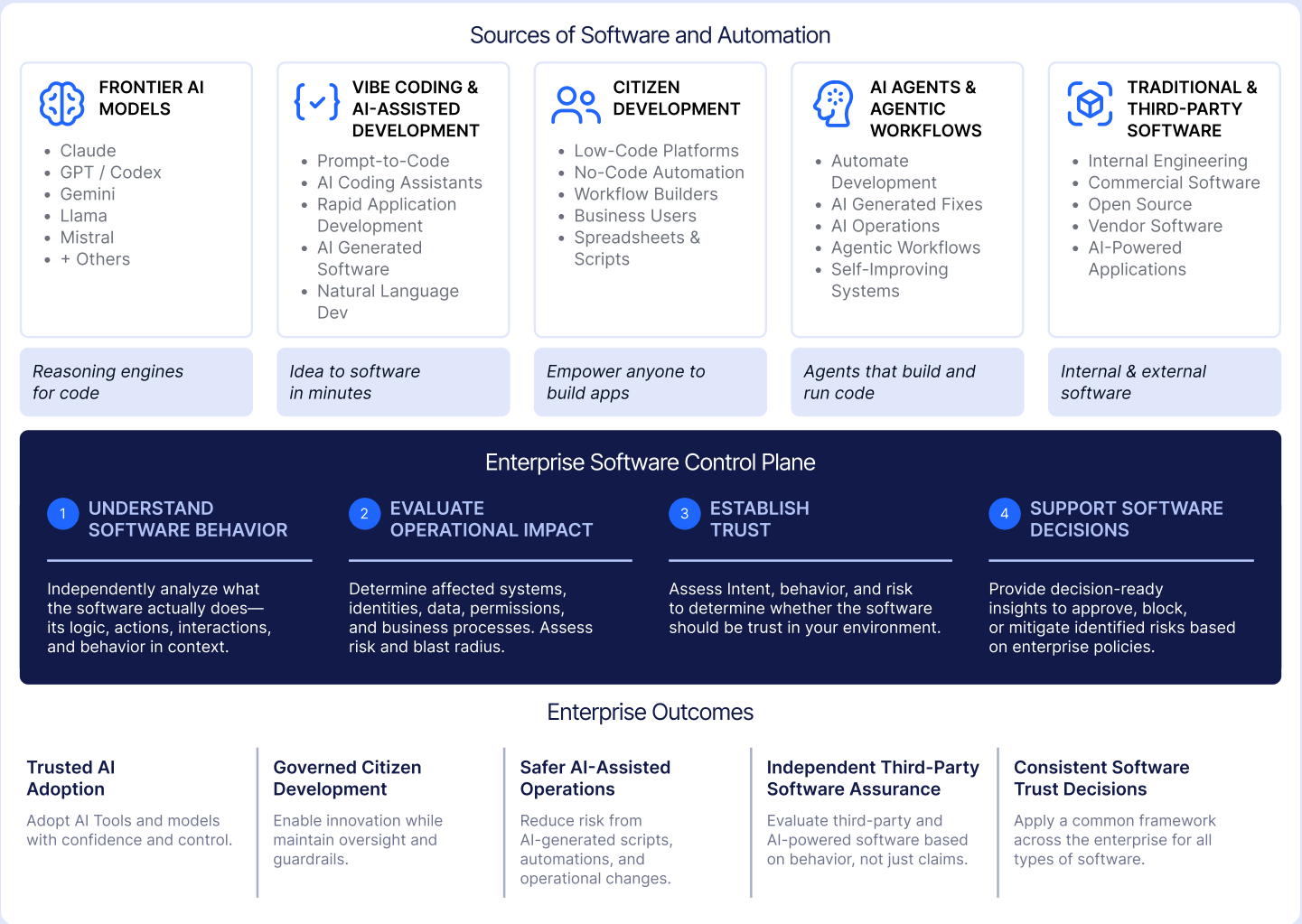


Introducing the Enterprise Software Control Plane

Existing controls provide valuable information about software risk, but organizations still need independent evidence of how software is likely to behave before it is installed, deployed, or executed.

This requires understanding software behavior, operational impact, dependencies, permissions, and previously unknown risks—regardless of how the software was created, who created it, or where it originated.

The Enterprise Software Control Plane provides a consistent layer for generating this behavioral evidence and supporting software decisions across the enterprise.





The Enterprise Software Control Plane establishes an independent, evidence-based source of truth for software and enables evaluation against enterprise risk frameworks before software is installed, deployed, or executed.

Software may originate from different sources and follow different review processes, but each ultimately requires a decision about whether it should be permitted to operate within the enterprise environment.

The Enterprise Software Control Plane operates alongside existing security, governance, and compliance capabilities, providing a common mechanism for applying enterprise requirements consistently across software sources and delivery models.

What the Enterprise Software Control Plane Does

The Enterprise Software Control Plane provides a common decision layer across the software lifecycle. It operates alongside existing development, security, software distribution, endpoint, and risk management systems, providing independent assessment as software enters the enterprise and as that software changes over time.

The Control Plane is designed to operate consistently regardless of how software was created, who created it, or where it originated. Software observed across enterprise distribution and transit points can be evaluated using the same underlying assessment model.

At its core, the Enterprise Software Control Plane performs four functions:

1

Observe software across the enterprise.

Establish visibility at key software distribution and transit points where software enters, moves through, and is deployed within enterprise environments.

2

Understand software behavior.

Independently assess what software does, including its dependencies, permissions, communications, operational impact, and previously unknown behaviors.

3

Evaluate software against enterprise requirements.

Apply behavioral assessment results across multiple enterprise risk frameworks, policies, regulatory requirements, and organizational objectives.

4

Support and sustain software decisions.

Provide decision-ready evidence to approve, block, or mitigate identified risks, with continuous reassessment as the software and its operating context change.

This creates a consistent decision architecture for software that may otherwise be evaluated through separate teams, processes, and technologies.



Why a Control Plane?

The Enterprise Software Control Plane applies this model to software, providing consistent visibility, behavioral assessment, risk framework evaluation, and decision support across the many channels through which software enters and changes within the enterprise.

Automating Governance Across Risk Frameworks

Governance frameworks define enterprise requirements related to security, compliance, operational resilience, software supply chain risk, AI usage, cryptographic readiness, and other business objectives. The challenge is applying those requirements consistently and at scale across an increasingly diverse software ecosystem.

A single software package may be evaluated by security teams, compliance teams, procurement functions, AI governance programs, and operational stakeholders. Each group may view the software through a different perspective and assess different aspects of risk. As the volume of software continues to increase, manually applying these requirements becomes increasingly difficult to scale.

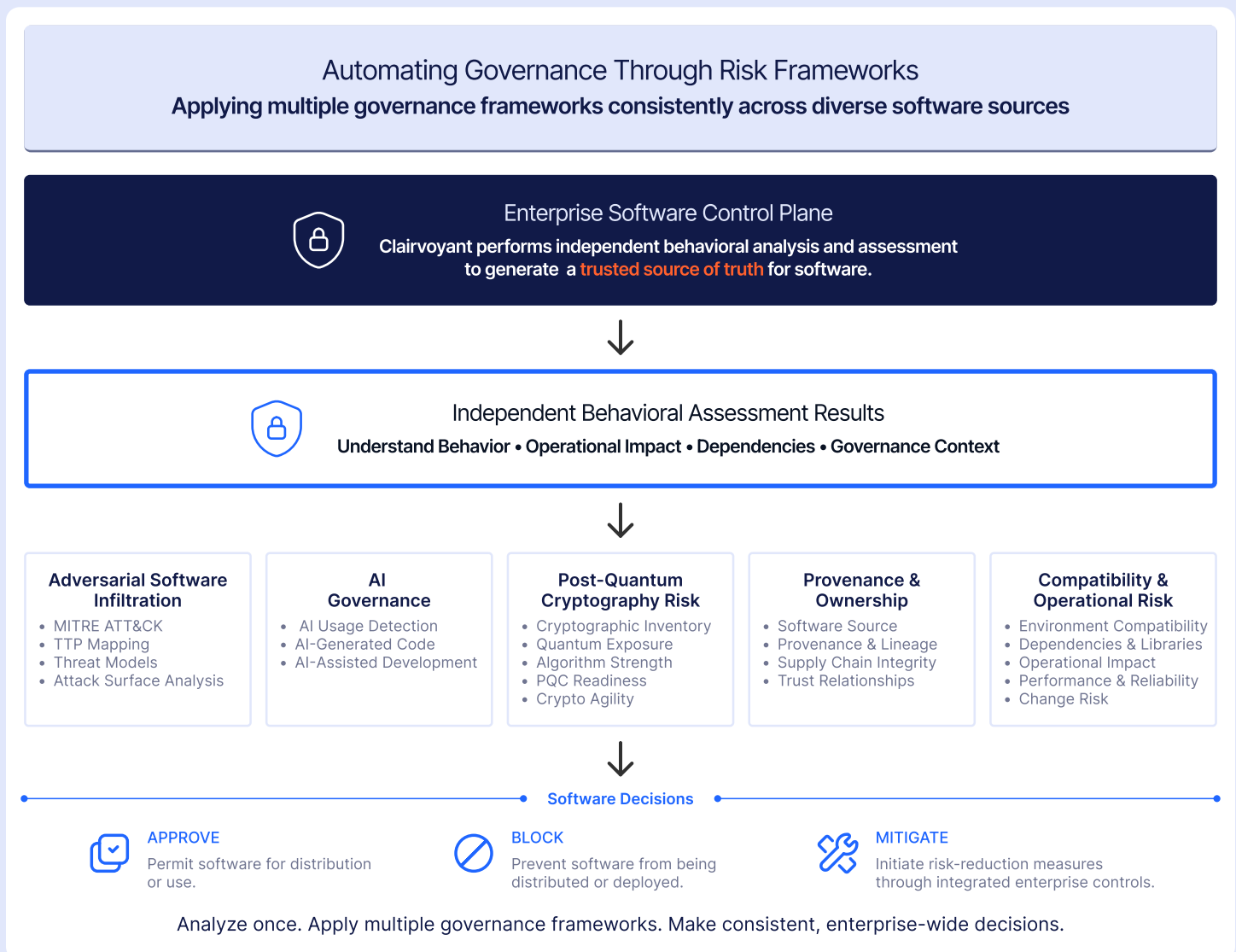
In many situations, this behavioral assessment may never occur. Software updates are routinely deployed, AI-generated code is incorporated into applications, shadow downloads enter enterprise environments, and third-party software is installed based primarily on supplier reputation, functionality, or operational need. Organizations are often required to make governance decisions without independently evaluating how software is likely to behave once executed.



The Enterprise Software Control Plane addresses this challenge through Clairvoyant Risk Frameworks. These predefined frameworks enable organizations to evaluate independent software assessment results against specific governance objectives and operational requirements.

These frameworks support software governance across domains such as software infiltration, AI risk, post-quantum cryptography exposure, provenance, and operational compatibility while using a common underlying software assessment.

This allows a single independent behavioral assessment to support multiple governance decisions across the enterprise.



For example, a security team may evaluate software through an Adversarial Software Infiltration Framework aligned with MITRE ATT&CK techniques and known attacker behaviors. A software governance team may evaluate the same software through the Provenance Framework to understand ownership, source, and trust relationships. Cryptographic teams may assess post-quantum cryptography exposure, while AI governance programs may evaluate AI-generated code, AI-assisted development practices, or AI-enabled functionality.

Each Clairvoyant Risk Framework applies the same independent behavioral assessment results to a different enterprise objective. Organizations can therefore make consistent decisions across multiple risk frameworks without repeating the underlying software assessment. As software creation continues to accelerate, Clairvoyant Risk Frameworks provide a scalable mechanism for evaluating software consistently across multiple enterprise requirements.

Integration and Operational Workflows

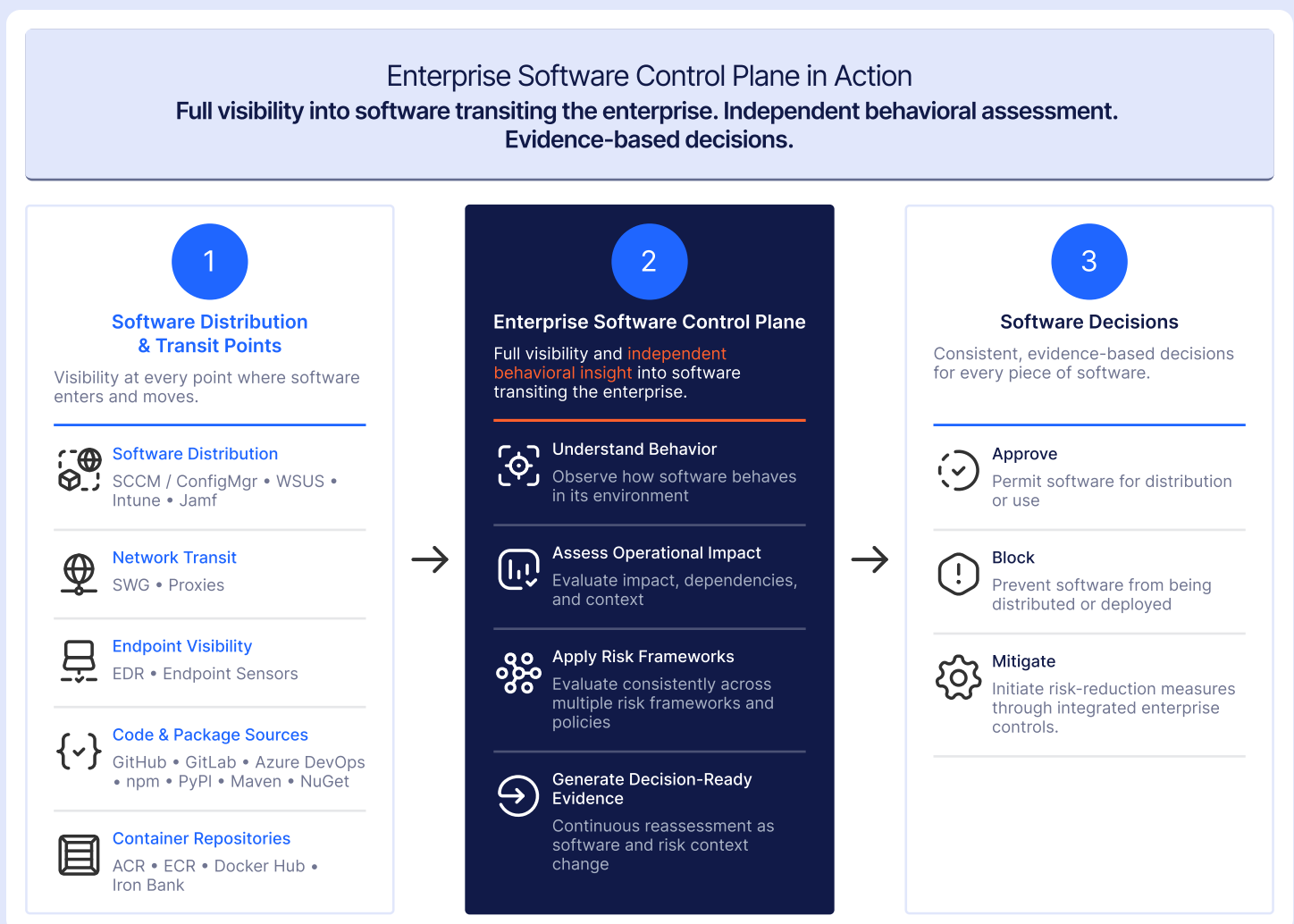
The Enterprise Software Control Plane integrates at key points where software enters, moves through, and is distributed across enterprise environments. These software distribution and transit points provide visibility into software before and during deployment without requiring organizations to replace their existing software delivery infrastructure.

Integration points can include enterprise software distribution platforms, network transit points, endpoint visibility platforms, code and package sources, and container repositories. Examples include SCCM/ConfigMgr and WSUS, Microsoft Intune, Jamf, secure web gateways and proxies, EDR or endpoint sensors, GitHub and GitLab, and ACR and ECR.



Software observed through these integration points can be independently assessed for behavior, operational impact, dependencies, and applicable enterprise risk frameworks. The resulting evidence supports consistent software decisions to approve, block, or mitigate identified risks.

These decisions can be operationalized through integrations with enterprise management and enforcement systems, allowing existing controls to carry out the required action.



Conclusion

Organizations increasingly evaluate software originating from a wide range of sources, including third-party suppliers, open-source ecosystems, AI-assisted development, citizen development initiatives, software updates, and software operating outside established governance processes.

Existing security, governance, procurement, and compliance processes provide important signals and controls across vulnerabilities, suppliers, compliance requirements, policy, package risk, and endpoint activity. They do not consistently provide insight into software behavior, operational impact, dependencies, permissions, or previously unknown risks before software is deployed.

The Enterprise Software Control Plane operates alongside these existing processes, providing independent software assessment and behavioral analysis. Assessment results can be evaluated against multiple enterprise risk frameworks, enabling software decisions to be applied consistently regardless of how software is created, selected, distributed, or introduced into the enterprise.

As software creation, updates, and distribution continue to accelerate, organizations require governance processes that can scale with the volume and speed of change. The Enterprise Software Control Plane provides an independent behavioral evidence foundation for applying enterprise risk frameworks consistently across an increasingly diverse software ecosystem, regardless of how software is created, selected, or delivered.

Additional implementation guidance can be found in Clairvoyant's accompanying Solution Guides, which examine specific governance workflows, risk frameworks, and deployment scenarios in greater detail.



References

The following publications, standards, and publicly disclosed industry research informed the perspectives presented in this paper.

Clairvoyant Intelligence Publications

Clairvoyant Intelligence. (Part 1) What We Run But Cannot See: Rethinking Trust in Software. 2026.

Clairvoyant Intelligence. (Part 2) The Collapse of Implicit Trust. 2026.

Standards and Frameworks

National Institute of Standards and Technology (NIST). Artificial Intelligence Risk Management Framework (AI RMF 1.0). January 2023.

MITRE Corporation. MITRE ATT&CK® Framework.

National Institute of Standards and Technology (NIST). Post-Quantum Cryptography Standards (FIPS 203, FIPS 204, and FIPS 205). August 2024.

Industry Research

Anthropic. Recursive Self-Improvement. 2026.

Industry Examples

Notepad++. Notepad++ Supply Chain Compromise / Update Infrastructure Hijacking. 2026.

Microsoft Defender Security Research Team and Microsoft Threat Intelligence. From Package to Postinstall Payload: Inside the Mastra npm Supply Chain Compromise by Sapphire Sleet. 2026.

Jamf Threat Labs. From ClickFix to Code Signed: The Quiet Shift of MacSync Stealer Malware. December 2025.

BerriAI / LiteLLM. Security Update: Suspected Supply Chain Incident. March 2026.

Palo Alto Networks Unit 42. OpenClaw's Skill Marketplace and the Emerging AI Supply Chain Threat. 2026.

