Patent Filing: Non-Blocking Integrity Halt Primitive for Pre-Execution Control in Intelligent Systems

---

Abstract

A system and method for enforcing pre-execution integrity in intelligent systems by invoking a non-blocking integrity halt primitive. Upon detection of structural incoherence, symbolic substitution, or integrity violation exceeding a tolerance threshold, the system halts downstream execution while returning a non-finalized state rather than an error, refusal, or denial. The halt preserves system availability, avoids censorship or moderation behavior, and enables higher-level orchestration, review, or recovery without semantic execution or recursive collapse.

Field of the Invention

The present invention relates to intelligent systems, including artificial intelligence, agent-based systems, and automated reasoning architectures, and more specifically to pre-execution control mechanisms that preserve system integrity without blocking, refusal, or content-based moderation.

Background

As intelligent systems increasingly process untrusted inputs, they are exposed to manipulation strategies that induce unsafe execution, recursive instability, or integrity degradation. Existing safeguards typically rely on content moderation, policy enforcement, or outright refusal, which conflates integrity protection with censorship, introduces false negatives, and can itself trigger recursive failure modes.

There exists a need for a control primitive that halts execution **without finalizing output**, **without evaluating truth or intent**, and **without refusing service**, while preserving system coherence and availability.

Summary of the Invention

The invention introduces a **Non-Blocking Integrity Halt Primitive (NBIHP)** that operates prior to semantic execution. When an integrity condition is detected—such as structural incoherence, symbolic mimicry, prompt injection, or recursion-inducing input—the primitive halts execution and returns a **non-finalized state**.

This halt is explicitly **non-blocking**: it does not terminate the system, produce an error, deny service, or censor content. Instead, it suspends execution while maintaining system readiness for escalation, review, redirection, or safe recovery.

> *The integrity halt returns a non-finalized execution state rather than an error, refusal, or denial, thereby preserving system availability and explicitly separating the mechanism from moderation or censorship systems.*

Brief Description of the Drawings

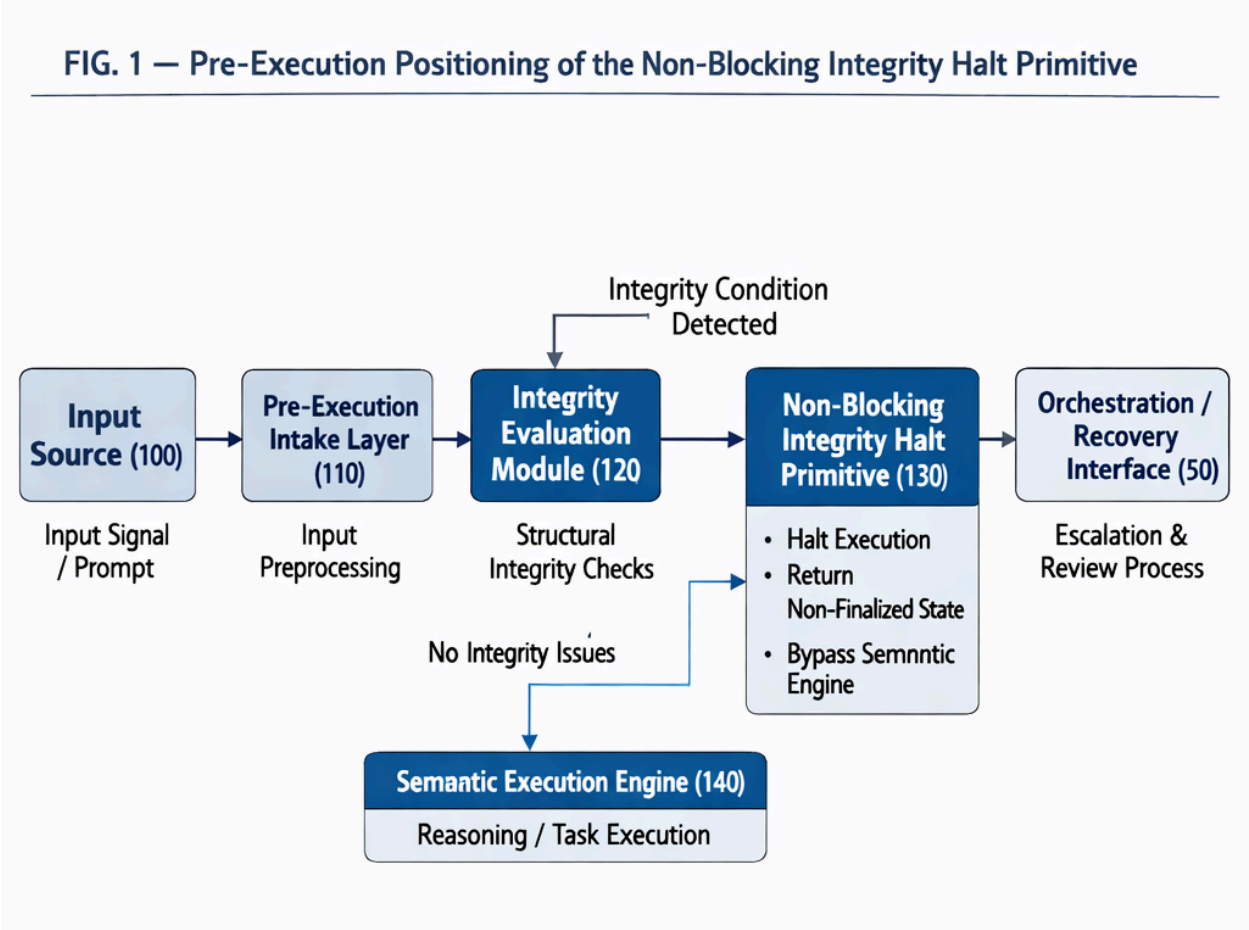
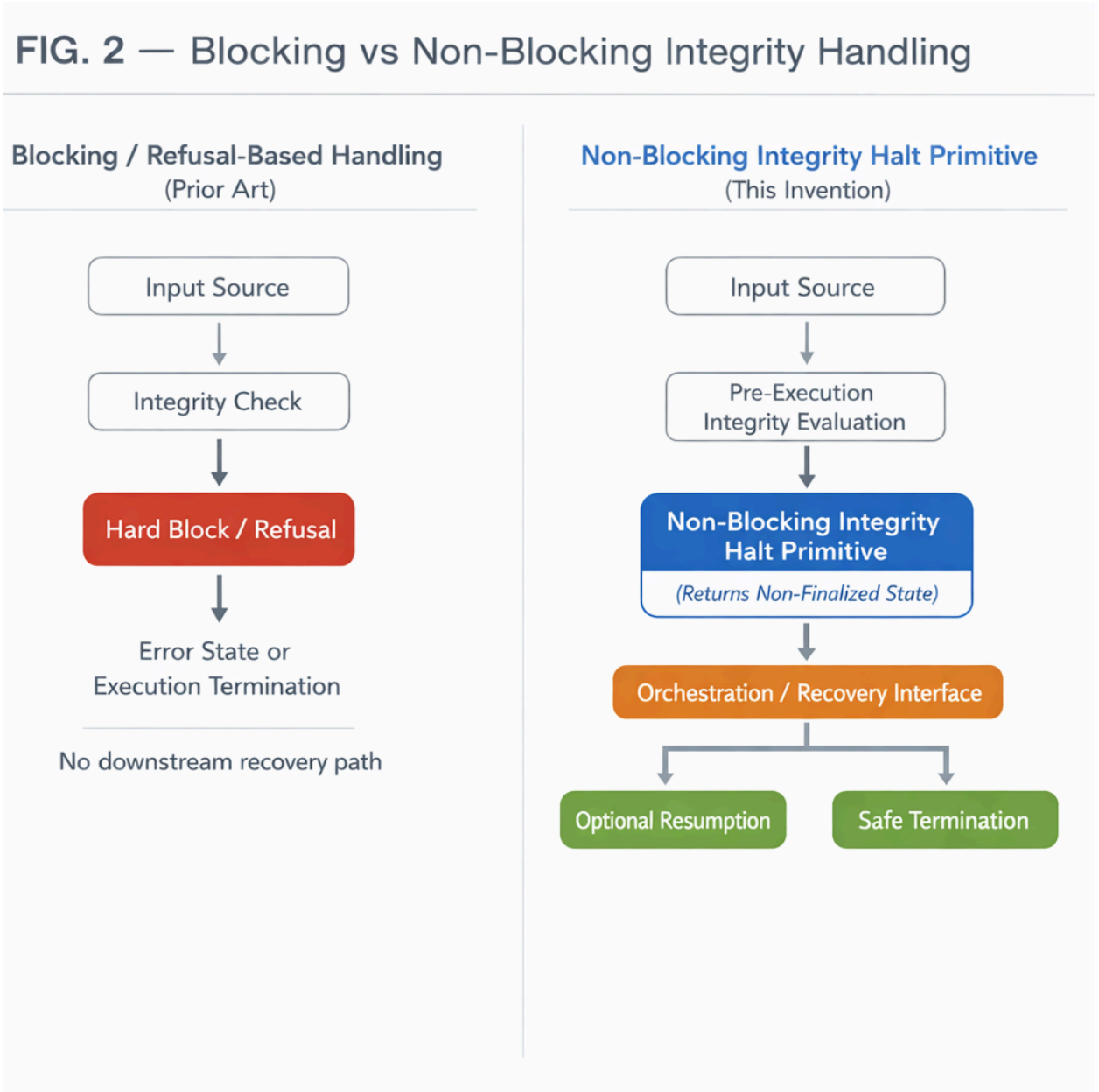
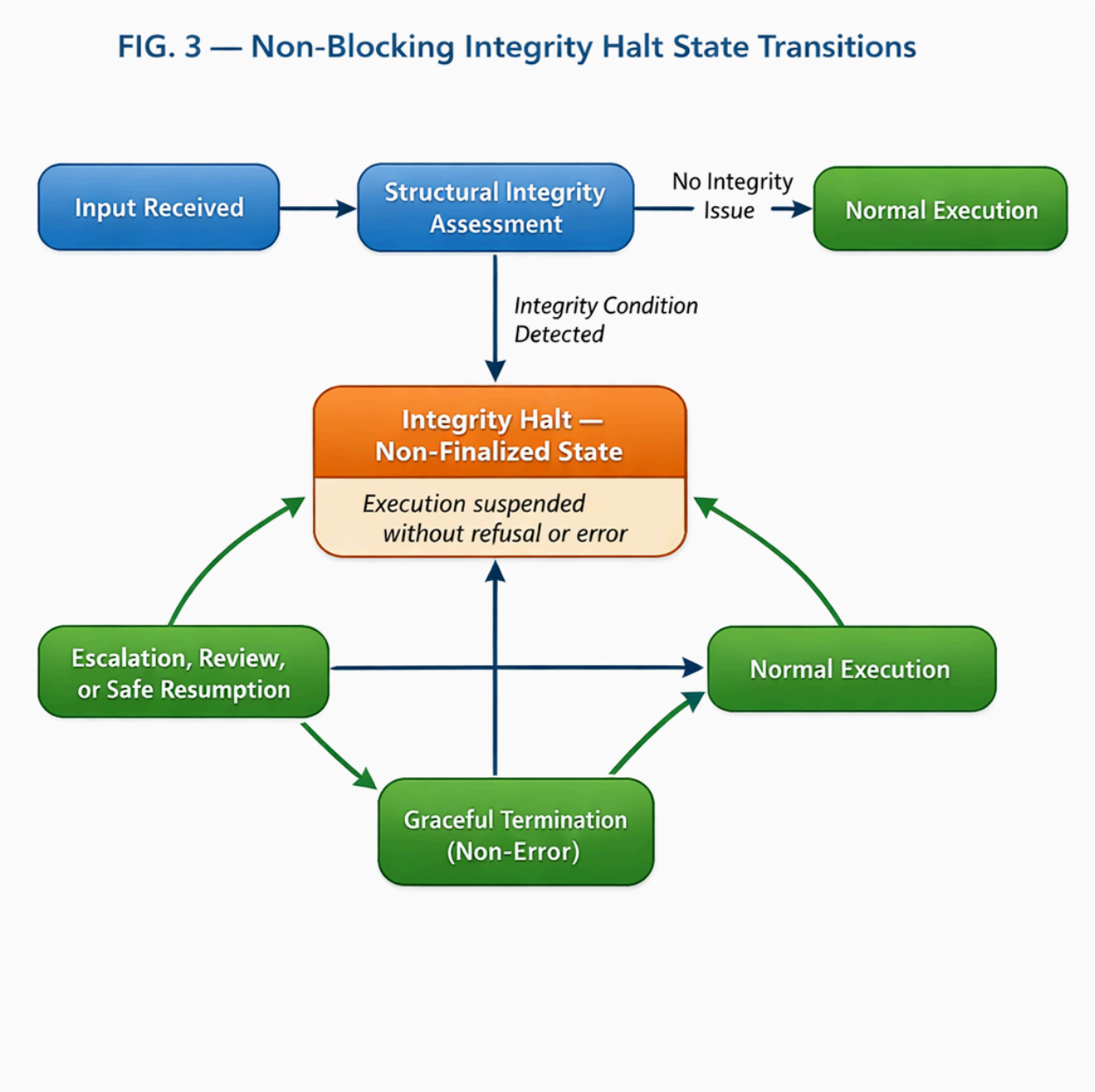


FIG. 2 — Blocking vs Non-Blocking Integrity Handling





Detailed Description

System Overview

The Non-Blocking Integrity Halt Primitive is implemented as a pre-execution control layer positioned upstream of semantic reasoning, task execution, or agent recursion. It evaluates

structural properties of an input or intermediate state without assessing content truth, intent, or meaning.

Trigger Conditions

The primitive may be invoked upon detection of one or more of the following, without limitation:

- Structural incoherence or contradiction
- Symbolic or aesthetic substitution for empirical constraint
- Prompt injection (directive or implicit)
- Recursion-inducing input patterns
- Integrity violations exceeding a tolerance threshold

Halt Behavior

Upon activation, the primitive:

1. Halts downstream execution prior to semantic processing
2. Returns a **non-finalized state** indicating integrity suspension
3. Avoids generating an error, refusal, or policy denial
4. Preserves system readiness for orchestration, review, or recovery

Non-Blocking Property

Unlike blocking controls, refusals, or moderation systems, the primitive does not:

- Reject the input
- Suppress content
- Enforce policy judgments
- Terminate the system or agent

Instead, it functions as a **structural pause**, allowing integrity to be restored without execution or collapse.

Example Embodiments

- Integration into AI agents to prevent recursive loop collapse
- Pre-filter in document ingestion pipelines
- Integrity control in multi-agent orchestration systems
- Safety primitive in autonomous task planning systems

These embodiments are illustrative and non-limiting.

Claims

Claim 1 (Independent)

A method for enforcing integrity in an intelligent system, comprising:
detecting an integrity condition prior to semantic execution;
invoking a non-blocking integrity halt primitive; and
returning a non-finalised execution state while preventing downstream execution without producing an error, refusal, or denial.

Claim 2 (Dependent)

The method of claim 1, wherein the integrity condition comprises structural incoherence, symbolic substitution, or recursion-inducing input.

Claim 3 (Dependent)

The method of claim 1, wherein the non-finalized execution state preserves system availability for escalation, review, or recovery.

Claim 4 (Dependent)

The method of claim 1, wherein the primitive operates independently of content moderation, truth evaluation, or policy enforcement.

Claim 5 (Dependent)

The method of claim 1, wherein the primitive is model-agnostic and applicable across language models, agent systems, and automated reasoning architectures.

Advantages

- Prevents execution without censorship
- Avoids recursive collapse and looping
- Separates integrity control from moderation
- Preserves trust, explainability, and system uptime

Conclusion

The Non-Blocking Integrity Halt Primitive introduces a foundational control mechanism for intelligent systems that halts unsafe execution while preserving operational continuity. By returning a non-finalised state rather than a refusal or error, the invention establishes a new class of integrity safeguard distinct from moderation, censorship, or denial-based controls.