A Petri Net-Based Framework for Automated Regulatory Compliance in Autonomous Systems Using Large Language Models

Aziz Amari
University of Carthage, Tunis, Tunisia
aziz.amari@insat.ucar.tn

***Abstract*— Ensuring regulatory compliance is a critical challenge for autonomous systems. Current verification methods rely either on rigid formal ontologies that require excessive manual overhead and continuous expert updates, or on purely generative Large Language Models (LLMs) that lack topological memory and are prone to dangerous semantic hallucinations. In safety-critical environments like aerospace, neither is sufficient. In this paper, we bridge this gap by treating textual regulations as formal, state-transition systems. We introduce an automated framework that maps unstructured legal text into an executable Petri net. This graph structure is exceptionally well-suited for modeling autonomous operations, as regulatory milestones natively map to Petri net Places (states) and compliance actions map to Transitions. By leveraging the mathematical boundaries of the graph to restrict generative outputs and utilizing a forward simulation reasoning engine, the language model is constrained to formal, logic-bound paths, drastically reducing hallucinatory errors. While our methodology is domain-agnostic, we validate our pipeline using the navigation and end-of-life disposal of spacecraft as a case study. By parsing complex international space debris mitigation guidelines, we demonstrate that this synergy of Petri nets and foundation models results in reliable compliance verification for next-generation autonomous operations.**

I. Introduction

Regulatory compliance across industries is characterized by its complexity, ambiguity, and necessity. Legal texts are written in natural language, often relying on semantic nuance and cross-references that make them difficult to formalize into strict mathematical or computational frameworks [1]. In safety-critical domains, the gap between the intent of a regulation and its implementation can lead to catastrophic failures. This challenge is highly acute in the governance of autonomous systems such as self-driving vehicles, industrial robotics, and spacecrafts, which operate in dynamic environments where strict adherence to operational constraints is mandatory. While our proposed methodology generalizes to the broad governance of autonomous agents, we showcase our solution using the navigation of satellites as a representative, safety-critical case study. As orbit becomes congested, adherence to multi-jurisdictional space debris mitigation guidelines (e.g., ISO-24113, FCC rules) is paramount [2].

Existing approaches to automated compliance largely fall into two categories: manual review and static formalization. Static formalization involves translating regulations into hard-coded ontologies [3]. While verifiable and rigorous, this method is labor-intensive and requires continuous manual expert intervention to maintain as regulations evolve. Conversely, Large Language Models (LLMs) have recently demonstrated proficiency in legal reasoning [4]; however, their stochastic nature poses severe risks for dynamic autonomous operations. Purely generative models are unconstrained, lack persistent topological memory, and are prone to dangerous semantic hallucinations when reasoning over long, multi-step constraints [5]. In aerospace compliance, where skipping a single mandated passivation step can result in explosive in-orbit fragmentation [6], "mostly correct" generation is unacceptable. A verifiable bridge must be forged between the semantic processing capabilities of foundation models and the strict bounds of formal symbolic systems [7].

To address this gap, we propose an automated framework that treats complex text as a state-transition system. Inspired by emerging theoretical paradigms such as the "Petri Net of Thoughts" (PNoT) [8], we argue that regulations for autonomous agents are best modeled not as static rules, but as dynamic processes [9]. Petri nets are exceptionally suited for this task due to their inherent ability to mathematically model concurrent states, sequential dependencies, and resource constraints, which perfectly mirror the prerequisites and mandatory actions found in regulatory text. In our framework, the LLM acts as a firing engine; navigating the strict mathematical boundaries of an executable Petri net [10] to restrict generative tendencies to formal, logic-bound paths.

The main contributions of this paper are threefold. First, we outline an *automated formalization pipeline* that translates unstructured regulatory text into a formal Petri net structure with defined Places for system states and Transitions for mandatory actions. Second, we implement a *forward simulation reasoning engine* where an LLM navigates the Petri net, creating a verifiable chain of thought mapped directly to regulatory clauses. Finally, we demonstrate *validation on historical spacecraft*, showing the system's ability to extract rules and detect non-compliance in realistic space debris mitigation scenarios.

The remainder of this paper is organized as follows. Section II details the regulatory dataset and mathematical foundations of our state-transition model. Section III explores the architecture of the Automated Formalization Pipeline. Section IV presents the resulting regulatory network, the Formal Execution Engine, and validation against historical spaceflight failures. Finally, Section VI concludes the paper with directions for future continuous-telemetry integrations.

II. Problem Formalization and Data

This section outlines the foundations and regulation documents utilized in this paper. We first introduce how to model a regulation as a formal Petri network. Subsequently, we detail the selected international regulatory documents, the generated mission event logs, and the historical fragmentation data cases used to evaluate the system.

A. Modeling Regulation as a Formal Petri Network

Regulations are frequently viewed as lists of constraints. However, regulatory adherence is a dynamic process. To alleviate the severe engineering bottleneck of manually verifying operational activities against text-based legal documents, we model this domain using a state-transition graph known as a Petri net $N = (P, T, A, M_0)$. The network consists of *Places* (P) representing the system's operational milestones or conditions (e.g., *Orbit_Licensed*), *Transitions* (T) representing autonomous events, planning actions, or physical maneuvers, and *Tokens* (M) representing the specific entities or resources navigating the processes. A regulatory clause such as "Spacecraft must be passivated after end of mission" translates to a formal topology where a token residing in $P_{\text{End\_Mission}}$ enables the mandatory transition $T_{\text{Passivate}}$, which upon firing deposits a token into $P_{\text{Safe\_State}}$.

B. Regulatory Documents

To select representative documents, we established criteria ensuring coverage of international guidelines, foundational industry standards, and enforceable national regulations that collectively dictate the lifecycle of an autonomous spacecraft. The foundation of the compliance checking framework relies upon a comprehensive corpus of international and domestic space safety guidelines. To ensure the framework accurately reflects modern compliance requirements, we curated a dataset of five foundational aerospace regulatory frameworks, detailed in Table I.

This corpus includes the foundations for orbital safety [11], the FCC 47 CFR Part 25 for national-level U.S. enforcement, and the UN COPUOS guidelines [12] supported by the European ECSS-U-AS-10C Rev.2 standard.

TABLE I
SELECTED REGULATORY DOCUMENTS

Document	Role in Framework
FCC 47 CFR Part 25	US satellite communication regulations and debris mitigation plans.
UN COPUOS Guidelines	International consensus on space debris mitigation principles.
ECSS-U-AS-10C Rev.2	European space sustainability and adoption of ISO standards.
COPUOS A/AC.105/C.1	Recent legal frameworks and national implementation reports [12].
AC105 (ESA/ECSS)	Foundational industry standards for debris mitigation.

C. Generated Mission Event Logs

To evaluate the framework against real-world scenarios, we synthesized 150 programmatic mission event logs based on historical fragmentation data documented in ESA Space Environment Reports [6]. These logs are structured as sequential arrays of contextual actions and satisfied preconditions (Figure 1). They are divided equally ($N = 50$) among three operational archetypes, detailed in Table II, representing both compliant missions and specific logical failures (e.g., delayed passivation or intentional breakup) that caused historical debris disasters. This approach provides the operational ground truth necessary to validate the engine's strict structural constraints outlined in Section V.

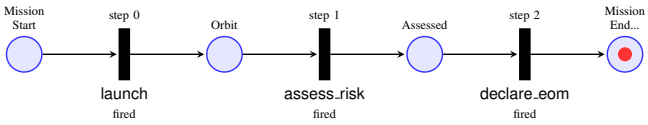

Fig. 1. Visual representation of a generated compliant mission event sequence.

TABLE II
SYNTHETIC MISSION EVENT LOGS ($N = 150$)

Profile	Description
Compliant Mission	Standard sequence ending in successful passivation.
Passivation Failure	Sequence skipping energy depletion before disposal.
Intentional Breakup	Abrupt spacecraft destruction without EOM declaration.

III. Methodology

The proposed framework operates as a five-stage automated pipeline that transforms regulatory documents into a formal, executable Petri net. As illustrated in Figure 2, this methodology comprises: Regulation Documents processing, Rule Extraction, Rule Formalization, Cross-Rule Synthesis, and final Petri Net Aggregation.

A. Rule Extraction

This step transitions from raw regulatory text to structured rules using a Large Language Model (LLM). We partition the text into chunks and pass them to Gemini 2.5 Pro. The LLM identifies modal sentences and extracts the core deontic requirements. As defined by the extraction prompt, each generated rule contains the clause_type (e.g., obligation, prohibition, permission), the responsible subject, the action verb, the target object, and any conditional triggers (conditions) or mathematical thresholds (quantitative_value).

B. Rule Formalization

We take the structured outputs from the previous step and, using a translation algorithm, construct Petri net subnets for each rule. The extracted features are programmatically translated into graph objects: triggering conditions

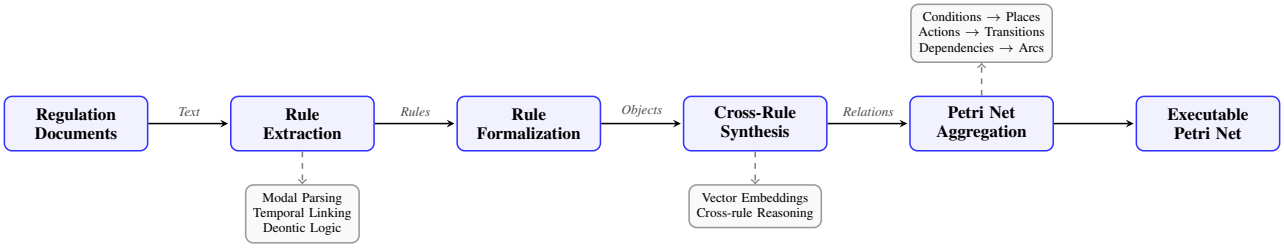

Fig. 2. A high-level conceptual summary of the automated formalization pipeline, extended to showcase the cross-rule semantic synthesis phase.

become Places, executable actions become Transitions, and temporal deadlines form constraint Arcs. Figure 3 illustrates a representative sub-net extracted from the FCC 20-54A1 regulation.

C. Cross-Rule Synthesis

Rules originating from different jurisdictions frequently intersect or overlap. To automatically detect and handle these intersections before compiling the final network, the parsed rules are encoded into a high-dimensional vector space using an embedding model. An agglomerative clustering algorithm then groups these rules based on vector similarity. Clusters are passed to the LLM to identify relationships as one of three structural modifiers: Equivalence ($\equiv$) where redundant rules share a single fulfillment transition, Dependency ($\rightarrow$) where a directed Arc is established from the source's fulfillment Place to the target's trigger Transition, and Contradiction ($\perp$) where a priority conflict Place is generated acting as an inhibitor arc. This step connects the fragmented sub-nets by injecting these dependency, inhibition, and synchronization arcs.

D. Petri Net Aggregation

In the final pipeline stage, the rule sub-nets and the relational arcs are compiled into a singular, globally executable Petri network.

To illustrate, consider the synthesis of overlapping international standards (Figure 4). Rule FCC 20-54A1 generically requires spacecraft to perform end-of-life disposal maneuvers, while UN COPUOS Guidelines explicitly mandate that spacecraft in Low Earth Orbit (LEO) must minimize their presence in the region to under 25 years. During aggregation, the synthesis agent identifies the hierarchical dependency between these texts. It bridges the two isolated sub-nets by injecting a directed structural sequence arc ($\rightarrow$) from the UN re-entry precondition to the FCC disposal maneuver transition. As a result, the aggregated global network mathematically guarantees that a spacecraft cannot legally execute its FCC disposal plan without first satisfying the rigorous threshold properties established by the UN guidelines.

E. Formal Execution and Forward Simulation

To clearly define our verification mechanics, it is important to distinguish this framework from traditional "token replay" techniques frequently used in process mining. Traditional token replay generally involves taking an existing, historical event log and tracing it backward or forward through a static model to measure conformance [13].

Conversely, our architecture acts as an active *formal execution engine* capable of *forward simulation* (or state-space traversal). Rather than just checking past logs, the fully aggregated Petri net calculates all legally enabled transitions given a present state marking. When an AI agent proposes a sequence of future actions (a "what-if" scenario), the execution engine fires these proposed transitions step-by-step. It advances the state of the network to mathematically deduce whether the sequence reaches a compliant goal state or triggers a blocked violation. This deterministic, forward-looking execution strictly bounds the generative capabilities of the LLM, ensuring its compliance answers are grounded in valid topological paths rather than probabilistic text generation.

IV. Resulting Network and Applications

This section details the resulting network and presents brief downstream demonstrations. The primary contribution of this work is the executable Petri net itself; the QA and verification applications presented hereafter serve to demonstrate its practical utility.

A. The Resulting Regulatory Network

From the ingestion of the five regulatory documents detailed in Section II, the semantic extraction engine initially parsed the raw text into structured JSON rules. The distribution of these extracted rules across the corpus is detailed in Table III.

TABLE III
Extracted Rules per Regulatory Document

Regulatory Document	Rules Extracted
FCC-20-54A1	342
AC105 (ESA/ECSS Standards)	76
ECSS-U-AS-10C Rev.2	14
UN Space Debris Mitigation Guidelines	12
Total Distinct Clauses	**444**

These rules were subsequently compiled into a unified Petri net [10], the statistics of which are summarized in Table IV.

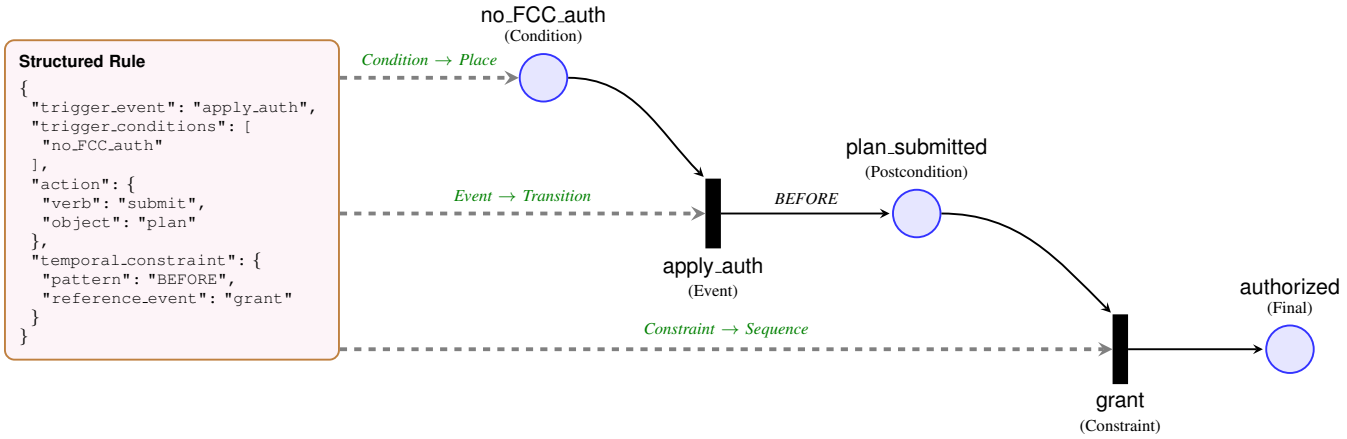

Fig. 3. Visualization of how a structured rule is translated into a Petri net.

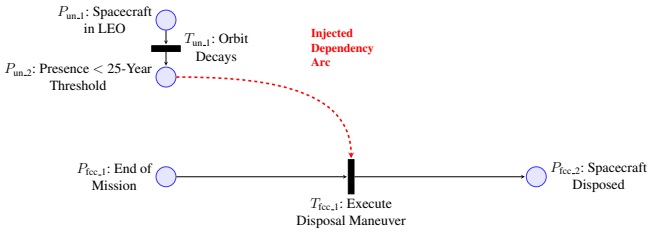

Fig. 4. Visualizing Petri Net Aggregation: A cross-rule sequence arc is injected to bind the FCC disposal maneuver to the UN 25-year criteria.

TABLE IV
GENERATED PETRI NET STATISTICS

Element	Count	Description
Places	1122	Distinct regulatory states and conditions
Transitions	741	Compliance actions and verifications
– Mechanical	∼600	Logic/Math checkable transitions
– Semantic	∼141	Transitions requiring LLM judgment
Arcs	1917	Logical connections representing flow

B. Reasoning-Based Governance Question Answering

The generated Petri net serves as a deterministic formal structure for reasoning-based governance Question Answering (QA). Rather than exposing the unstructured text directly, we architect a tool-calling workflow that leverages this formally structured network to answer complex regulatory queries. This workflow is driven by an autonomous QA agent that acts as an orchestration layer, securely mapping natural-language inputs to specific graph traversals. By strictly bounding the model to this mathematical structure, we ensure significantly higher confidence in the generated responses compared to standard Retrieval-Augmented Generation (RAG) pipelines, where lack of strict operational context often leads to unverified associations.

To interact with the graph, the QA agent utilizes a standardized tool list, allowing it to traverse relevant parts of the network and simulate them using the formal execution engine. These tools are detailed in Table V.

The architecture of this governance agent operates in four

TABLE V
QA AGENT SEMANTIC AND FORMAL TRAVERSAL TOOLS

Tool/Method	Functionality
identify_domains	Dynamically routes queries to regulatory subnets via dense vector embeddings.
find_relevant_rules	Retrieves specific formal graph constraints matching the query semantics.
simulate_scenario	Executes an LLM-proposed what-if sequence through the formal execution engine.
get_citation	Bi-directionally maps formal rule results back to the original unstructured legal text.
get_thresholds	Retrieves quantitative constraints (e.g., altitude limits) bound to the active states.

distinct phases (Figure 5). First, **Vector-Based Retrieval** dynamically maps the user's natural language question to the most relevant mathematical subnets using the same embedding model utilized during document ingestion, ensuring robust, multi-domain context retrieval. Second, **Simulation-Augmented Reasoning** tasks the LLM to propose a logical sequence of transition actions (a what-if scenario) based on the user query and the retrieved regulatory states. Third, **Forward Simulation** passes these proposed operational actions through the mathematical PetriNetSimulator. The simulator tracks token movement via constraint-based execution, logging blocked transitions and strict regulatory violations. Finally, the logical execution state is fed back into the LLM for **Response Generation**, ensuring the answer is strictly grounded in an explicit operational capability test.

Because each transition in the Petri net retains a direct mapping to the original unstructured text via its *Clause ID*, we feed the source text and the execution log into the LLM. This provides the agent with definitive, grounded context during the formulation of the answer.

V. EXPERIMENTAL EVALUATION

To evaluate the utility of our approach, we conducted three experiments analyzing hallucination reduction, real-world telemetry conformance, and computational scalability.

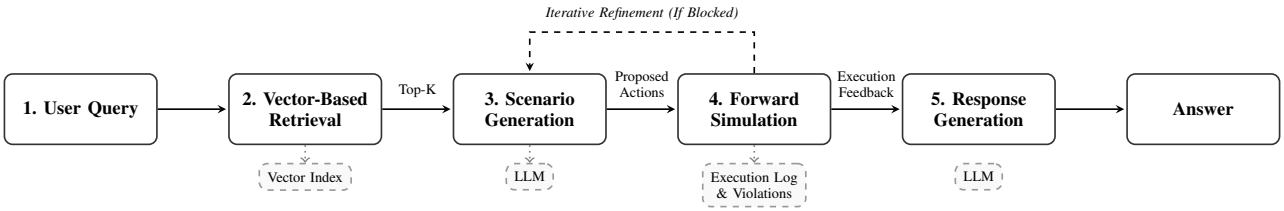

Fig. 5. Architecture of the Reasoning-Based Governance Engine. The LLM acts iteratively as an intelligent planner (Scenario Generation) and synthesizer, strictly bounded by the mathematical execution trace produced by the forward simulator.

A. Validation Dataset Construction

To facilitate an evaluation of our framework's capacity for compliance reasoning against complex textual narratives, we constructed a dataset comprising 150 synthetic mission event logs. This process was executed as a two-stage pipeline designed specifically to evaluate the extraction and formal verification of complex, unstructured narratives.

First, in the **Unstructured Generation Phase**, a Large Language Model (LLM) was prompted with historical mission archetypes (as detailed in Section II): fully compliant modern constellations, accidental passivation failures (e.g., Ariane V16), and intentional fragmentation events (e.g., Cosmos-1408). The model generated completely unstructured, highly variable natural-language telemetry transcripts, anomaly reports, and ground-station chronologies summarizing each mission's distinct lifecycle.

Second, in the **Semantic Extraction Phase**, the LLM parses the unstructured narratives to map observations into the formal parameters required by the Petri net. Specifically, the model extracts a float value for collision probability and three Boolean status flags: disposal_within_25y, passivated, and debris_release.

This two-stage approach is necessary because raw mission narratives cannot be directly evaluated by the mathematical Petri net. The second step bridges this gap by explicitly translating natural language semantics into the formal Boolean states required by the simulator. For example, a generated report stating "the satellite successfully lowered its orbit and depleted its propellant reserves" is deterministically mapped by the LLM into the required formal state: disposal_within_25y=True and passivated=True.

B. Experiment 1: Hallucination Reduction

A critical challenge when using LLMs in autonomous regulatory systems is the elimination of hallucinations during compliance verification. In this context, we define a "hallucination" as any evaluation output by the model that logically contradicts the deterministic ground-truth state of the mission profile. We established a baseline using a standard Retrieval-Augmented Generation (RAG) approach powered by a foundation model (Gemini 2.5 Pro). The baseline RAG agent was provided with the full vectorized context of the entire five-document corpus (Section II) and asked to evaluate the compliance of the generated dataset containing 150 historical mission profiles.

These profiles spanned three distinct archetypes ($N = 50$ each): compliant modern deployments (e.g., SpaceX Starlink [14], Planet Labs), passivation failures (e.g., Ariane V16, NOAA-16 [15]), and intentional fragmentation events (e.g., Cosmos-1408, Fengyun-1C [15]).

As shown in Table VI, the baseline RAG system struggled with temporal state tracking, achieving correct structural conformance on 98 out of the 150 profiles. Therefore, the baseline produced 52 hallucinated evaluations. While the baseline successfully identified obvious violations in the ASAT breakup scenarios, it exhibited failures in temporal reasoning when tracing the preconditions of the historical passivation failures like Ariane V16.

When the identical 150 mission profiles were evaluated using our Formal Execution Engine, the system successfully analyzed all 150 profiles without experiencing a single hallucination (0 out of 150). Because the formal execution architecture physically restricts the LLM's scenario generation to valid graph traversals, logically impossible outputs become mathematically restricted by design. The governing pipeline deterministically modeled previous breakup events without relying on probabilistic recall [16], [17].

TABLE VI
COMPLIANCE EVALUATION SUCCESS: BASELINE RAG VS. OUR APPROACH ($N = 150$)

Historical Archetype ($N = 50$)	**Baseline**	**Formal Execution**
Compliant (Starlink/Planet)	46 / 50	**50 / 50**
Passivation Failure (Ariane/NOAA)	29 / 50	**50 / 50**
ASAT Breakup (Cosmos/Fengyun)	23 / 50	**50 / 50**
Total Success Configurations	**98 / 150**	**150 / 150**

C. Experiment 2: Simulated Telemetry Conformance Verification

To validate the system against simulated mission data, we deployed the exact same Forward Simulation Agent described above as an active run-time conformance checking agent evaluating historic telemetry profiles. We tested two scenarios: a compliant modern LEO constellation (e.g., SpaceX Starlink) and a historic non-compliant debris event (e.g., NOAA-16). As documented extensively in NASA and ESA analyses [16], [17], modeling historical breakup events through rigorous state constraints is critical for preventing future collisions.

The telemetry sequences used for this evaluation were manually abstracted from historical mission chronologies to isolate critical regulatory checkpoints. While simplified for clarity in this demonstration, these sequences represent the core state-transition logic required to evaluate high-level compliance under the UN and ECSS frameworks.

The compliant mission telemetry logged the sequence: The state-space traversal algorithm verified that all pre-

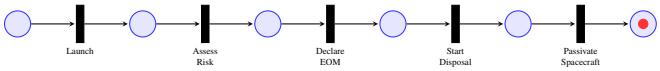

Fig. 6. Telemetry sequence for a fully compliant historical mission.

requisites were met in the correct sequence, returning a VALID compliance state. Conversely, the NOAA-16 profile (which historically skipped battery passivation leading to an in-orbit breakup event [16]) logged the sequence: The formal mathematical engine blocked the execution at

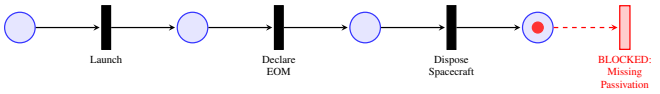

Fig. 7. Telemetry sequence for a non-compliant historical passivation failure resulting in a breakup event.

the dispose_spacecraft transition, correctly identifying that the mandatory passivate_spacecraft precondition had been skipped. This demonstrates the framework's ability to act as an unyielding, real-time "Space AI" [18] compliance auditor.

VI. Conclusion

We address the critical challenge of ensuring verifiable regulatory compliance in autonomous systems, a task where rigid formal ontologies often suffer from excessive manual overhead and the requirement for continuous expert updates, while standard Large Language Models (LLMs) remain prone to hallucination. To solve this, we presented a framework that automatically formalizes unstructured guidelines—such as space debris mitigation standards—into an executable hierarchical Petri net. While we demonstrated a reasoning agent as a sample application, the core contribution of this work is the automated network setup itself, which provides a flexible formal foundation for diverse regulatory verification tasks. By grounding the generative capabilities of foundation models within a strict, mathematical state-transition logic, our framework bridges the gap between natural language interpretation and deterministic safety.

Our experimental results demonstrate that this structure-enhanced architecture successfully captures complex temporal regulatory dependencies. When evaluated against historical mission telemetry, the forward simulation engine enforced strict hallucination reduction, bounding the LLM to mathematically valid states compared to standard Retrieval-Augmented Generation (RAG) baselines.

Despite these advantages, the engine's logical integrity remains structurally dependent on the quality and completeness of the initial *Rule Extraction* phase, where we rely on LLM-driven parsing to initialize the network. Future work must explore integrating automated theorem-proving [19] to formally verify this natural-language semantic translation. Furthermore, we intend to evaluate the approach across broader industries, such as healthcare or financial regulation, and perform dedicated quality assessments of the generated networks. Future research should also conduct direct comparisons between our automated framework and formal models meticulously crafted by human experts to benchmark performance and scalability. Ultimately, by addressing these limitations and scaling to handle competing multi-jurisdictional constraints, this approach paves the way for safe, self-regulating autonomous systems capable of deterministically navigating complex legal landscapes.

References

[1] X. Pan *et al.*, "Large language models for software engineering: A systematic literature review," *arXiv preprint arXiv:2308.10620*, 2023.

[2] S. C. Dola *et al.*, "Safety assurance of neuro-symbolic systems in aerospace," in *Digital Avionics Systems Conference (DASC)*, 2023.

[3] M. Kuhn *et al.*, "Formal verification of neural networks: A survey," *Annual Reviews in Control*, 2023.

[4] N. Guha *et al.*, "Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models," in *Advances in Neural Information Processing Systems*, vol. 36, 2023.

[5] E. Quevedo Caballero, "Leveraging large language models for legal document understanding," 2024. [Online]. Available: rivas.ai/pdfs/quevedo2024llms.pdf

[6] ESA Space Debris Office, "Esa space environment report 2023," European Space Agency, Tech. Rep., 2023. [Online]. Available: https://www.sdo.esoc.esa.int/environment_report/Space_Environment_Report_latest.pdf

[7] A. Li *et al.*, "Neuro-symbolic ai: An emerging class of ai workloads and their characterization," *arXiv preprint arXiv:2109.06133*, 2023.

[8] A. Gavric, D. Bork, and H. Proper, "Petri Net of Thoughts: A Structure-Enhanced Prompting Approach for Process-Aware Artificial Intelligence."

[9] T. Murata, "Petri nets: Properties, analysis and applications," *Proceedings of the IEEE*, vol. 77, no. 4, pp. 541–580, 1989.

[10] M. Figat *et al.*, "Ontology-driven robotic specification synthesis (rstm2)," *arXiv preprint arXiv:2602.05456*, 2026.

[11] FCC, "Mitigation of orbital debris in the new space age," Federal Communications Commission, Tech. Rep., 2020. [Online]. Available: https://docs.fcc.gov/public/attachments/FCC-20-54A1.pdf

[12] ILWR, *The Cologne Manual on Space Traffic Management*, 2023.

[13] W. van der Aalst *et al.*, "Replaying history on process models for conformance checking," 2012.

[14] SpaceX, "Spacex non-geostationary satellite system, attachment v: Orbital debris mitigation plan," Federal Communications Commission, Tech. Rep., 2020.

[15] N. Johnson *et al.*, "History of on-orbit satellite fragmentations," NASA Orbital Debris Program Office, Tech. Rep., 2008.

[16] E. S. D. Office, "Analysis of historical in-orbit breakup events and their mitigation," in *Proceedings of the 7th European Conference on Space Debris (SDC7)*, 2017. [Online]. Available: https://conference.sdo.esoc.esa.int/proceedings/sdc7/paper/1005/SDC7-paper1005.pdf

[17] M. Garcia *et al.*, "History of on-orbit satellite fragmentations (15th edition)," NASA Orbital Debris Program Office, Tech. Rep., 2019. [Online]. Available: https://ntrs.nasa.gov/api/citations/20190033947/downloads/20190033947.pdf

[18] A. Ancona *et al.*, "Space ai: Leveraging artificial intelligence for space to improve life on earth," *arXiv preprint arXiv:2512.22399v1*, 2025.

[19] L. Zhang *et al.*, "Selene: Pioneering automated proof in software verification," *arXiv preprint arXiv:2401.07663*, 2024.