What We Know We Do Not Know: Systematic Risk Identification from Contradictory Safety Documentation

Masoud Ebrahimi[1,2], Kaj Hänninen[2], Kristina Lundqvist[2], and Marjan Sirjani[2]
[1] *Formal Trust AI, Sweden. E-mail: masoud.ebrahimi@formaltrust.ai*
[2] *Department of Computer Science & Engineering, Mälardalen University, Sweden.*

Ensuring accountability in human-centered safety-critical systems, such as autonomous vehicles and medical devices, requires analysis methods that expose latent hazards implied by incomplete documentation. These systems must maintain safety, satisfy normative and ethical requirements, comply with regulations, and manage complex human–machine interactions. Existing methods fall short in verification of documented properties and attribution of responsibility; a key cause is the presence of undocumented safety-relevant predicates (known unknowns) that remain undetected. We propose a Kripke-based approach that connects the Open World Assumption of ontological knowledge graphs with Closed World Assumption verification by exhaustive enumeration of feasible completions. The approach exposes latent hazards that remain implicit under Open World Assumption validation of incomplete documentation. Our methodology addresses the lack of precision in the extraction of knowledge graphs from informal documentation, which prevents reliable automated validation and formal correctness guarantees. We construct Kripke structures whose states are models that satisfy knowledge graph configurations derived from potentially contradictory documentation. The framework incorporates expert feedback to align models with domain knowledge and regulatory expectations. A case study of a simplified automotive seatbelt warning system demonstrates how the framework compares documented safety properties with latent-hazard criteria. The case reveals a 40.9% coverage gap in safety classification: many states classified as safe by the specification nevertheless harbor latent hazards resulting from sensor–actuator conflation and unobservable safety properties.

Keywords: Unknown risks identification, Open-World Assumption, Known Unknowns, safety-critical systems, accountability and responsibility.

1. Introduction

Safety-critical systems are governed by requirements, design standards, operational procedures, and regulatory compliance artifacts (Alur, 2015; ISO, 2018). Such documentation is often incomplete, uses modal language, and can contain contradictory statements when interpreted in different operational contexts (Palshikar, 2001). Documentation management for safety analysis is challenging (Hall et al., 2017). The assignment of accountability further complicates the process.

Knowledge graphs structure and integrate documents from heterogeneous sources by representing entities as nodes and relationships as edges (Ehrlinger and Wöß, 2016). Knowledge graphs enable semantic queries, automated reasoning, and traceability while operating under the Open World Assumption, where the absence of a fact does not imply its negation (Gruber, 1993). However, knowledge graphs do not capture modality: they encode static relationships but not the modal distinctions that Kripke semantics requires for systematic state enumeration in safety analysis.

Constructing knowledge graphs from documentation poses three challenges: (i) modal statements can be misinterpreted as factual assertions, producing contradictions; (ii) documentation is often incomplete, leaving relationships unspecified; and (iii) the absence of a formal semantics prevents detection and mitigation of axiom violations.

Despite these challenges, knowledge graph construction from documentation is feasible using established ontology engineering and knowledge acquisition workflows that combine automated extraction with expert curation (Musen, 2015; Ji et al., 2022). In safety and security engineering, evidence structuring, traceability, and risk documentation practices are well established (Leveson,

Proc. of the 36th European Safety and Reliability Conference
Edited by José C. Matos, Paulo B. Lourenco, Michael Beer, Daniel Oliveira and Edoardo Patelli
Copyright © 2026 by ESREL2026 Organizers. *Published by* Research Publishing
ISBN: 981-973-0000-00-0 :: doi: 10.3850/981-973-0000-00-0_output

2012; Hall et al., 2017; Alur, 2015). Our contribution begins with these extracted triples and adds formal modal validation to expose latent hazards.

For certification, systems must satisfy safety properties under all normal and abnormal conditions. Therefore, safety analysis typically adopts the closed-world assumption (CWA), in which relevant facts are assigned truth values to enable formal verification (Clarke et al., 2018). The mismatch between OWA and CWA creates a verification gap that can hide latent hazards. When hazards arise from undocumented facts (known unknowns), attributing responsibility becomes unclear and some safety gaps may be overlooked.

We propose a construction-time reasoning methodology that bridges Open World Assumption and Closed World Assumption by validating knowledge graphs through exhaustive enumeration of feasible completions. Instead of treating contradictions as errors to be resolved, we represent them as distinct states to be enumerated. We interpret knowledge graphs as logical conjunctions of RDF triples. Starting from an initial state that contains only entity types, we incrementally add documented relationships and search for maximally determined states. A state is maximally determined if adding any additional relationship yields a contradiction.

In the resulting Kripke structure, we assign self-loops to these states to indicate that they are terminal (i.e., no further consistent extensions exist).

This construction partitions contradictory statements into distinct states, each representing a coherent interpretation of the documentation. For a state with an undecided relationship (a known unknown), we branch into two successor states: one in which the relationship holds and one in which it does not, thereby exposing latent hazards that were previously unspecified. The Kripke structure is complete once all reachable paths end in maximally determined terminal states with self-loops.

We annotate terminal states with safety properties as safe or unsafe. We quantify hazard severity as the minimal symmetric difference between the labeling of an unsafe terminal and the labeling of any safe terminal: small distances indicate isolated specification gaps, while large distances indicate systematic underspecification.

To preserve readability, we first define the formal objects and then interpret their practical meaning in the case study and discussion.

Section 2 covers preliminaries. Section 3 proposes our methodology, the Kripke semantics for knowledge graphs (KGs), and their validation. Section 4 presents a case study. Section 5 compares with related work, and Section 6 concludes and outlines future work.

2. Preliminaries

This paper brings together concepts from safety engineering, formal methods, and knowledge representation. To facilitate understanding across these disciplines, we first define key terms.

2.1. *Terminology*

System. This is a real-world entity being analyzed (for example, a medical infusion pump). A system comprises interacting components whose behavior is described in specifications.

Ontology. An ontology is a structured vocabulary that defines domain concepts, relationships, and governing constraints (see Definition 2.1).

Knowledge Graph. A knowledge graph is a graph representation that instantiates an ontology with facts extracted from documentation: nodes represent entities and edges represent relationships (see Definition 2.2).

Atomic Proposition. An atomic proposition is a Boolean fact that cannot be decomposed (for example, $\mathrm{poweredOn}(\mathrm{pump}_1)$).

State. This is a collection of atomic propositions with definite truth values. The term "state" corresponds to the possible-world concept in modal semantics; we use "state" throughout this paper.

Kripke structure. A directed graph whose nodes represent system states and whose edges represent additions of facts (atomic propositions or their negations). A Kripke structure represents the feasible search space of possible system designs (see Definition 2.3).

Terminal state. A maximally determined state in which adding any new fact would yield a contradiction. Terminal states have self-loops.

Model. A state that satisfies the constraints of a knowledge graph K; i.e., a collection of atomic propositions that conforms to the facts of K and to the governing constraints of the ontology.

Property. A logical assertion corresponding to required system behavior or safety criteria.

Specification. A description of required system behavior, system constraints, and system properties. They may be incomplete, contradictory, or ambiguous.

Validation. Establishing whether specifications correctly and accurately capture intended behavior and stakeholder requirements.

Verification. Establishing whether a system satisfies specified properties through formal proof or exhaustive analysis.

Hazard. A state with potential to cause harm, damage, or loss to people, property, or the environment. Hazards may arise from unsafe configurations, design flaws, or operational conditions.

Latent Hazard. A hazard that is not detectable under documentation-level (Open World Assumption (OWA)) reasoning because it depends on predicates left undecided by the documentation; it becomes explicit when feasible completions are enumerated and checked against a stated safety property.

Risk. The combination of harm likelihood and severity, quantifying exposure to hazardous configurations reachable during system operation.

Residual Risk. Risks remaining after mitigation, including accepted hazards and *known unknowns*; i.e., hazards exposed by formal analysis that arise from incomplete specification.

Assurance. The systematic process of providing justified confidence that a system operates safely under specified conditions, supported by evidence linking properties to requirements.

Coverage. The extent to which terminal states satisfy documented properties. Coverage gaps reveal configurations classified as safe yet harboring latent hazards.

Open World Assumption. The principle that unknown facts are not assumed false, allowing incomplete information without contradiction.

Closed World Assumption. The principle that unknown facts are assumed false, necessitating assignment of truth values to all relevant facts and enabling exhaustive verification.

Definition 2.1. An ontology $\mathcal{O} = \langle \mathcal{C}, \mathcal{R}, \mathcal{A} \rangle$ is a triple: a finite set of concepts $\mathcal{C}$, a finite set of relations $\mathcal{R}$, and a finite set of axioms $\mathcal{A}$ that define constraints over concepts and relations.

Running Example 1. For a medical infusion pump, the ontology $\mathcal{O}$ defines classes $\mathcal{C}$, relations $\mathcal{R}$, and axioms $\mathcal{A}$ as follows:

$\mathcal{C} = \{\mathrm{Pump}, \mathrm{IVLine}, \mathrm{Battery}, \mathrm{Alarm}\}$

$\mathcal{R} = \{\mathrm{alarmOn}, \mathrm{isConnected}, \mathrm{isInfusing}, \mathrm{lowBattery}, \mathrm{poweredOn}\}$

$\mathcal{A} = \{\mathrm{A1} : \mathrm{isInfusing}(x) \rightarrow \mathrm{poweredOn}(x)\}$

meaning "A1. *infusing requires power.*"

Definition 2.2. Given an ontology $\mathcal{O} = \langle \mathcal{C}, \mathcal{R}, \mathcal{A} \rangle$, a knowledge graph $K = \langle N, E, \ell \rangle$ is defined by: N is a finite set of nodes (entities); $E \subseteq N \times N$ is a set of directed edges (relationships); and ℓ labels nodes with concepts ($\ell : N \rightarrow 2^{\mathcal{C}}$) and edges with relations ($\ell : E \rightarrow 2^{\mathcal{R}}$). All instances and relationships in K must satisfy the axioms $\mathcal{A}$.

Definition 2.3. A Kripke structure over atomic propositions $\mathbb{AP}$ is the tuple $M = \langle Q, I, R, L \rangle$. Here Q is a finite set of states, $I \subseteq Q$ is a non-empty set of initial states, $R \subseteq Q \times Q$ is a transition relation, and $L : Q \rightarrow 2^{\mathbb{L}}$ is a labeling function. We write $\mathbb{L} = \mathbb{AP} \cup \{\neg p \mid p \in \mathbb{AP}\}$ for the set of literals.

State terminology. In this context, a *state* is an element $q \in Q$ of the Kripke structure M. In knowledge-representation literature, the term "world" corresponds to this notion of state; we use "state" throughout this paper.

Running Example 2. Suppose a Kripke structure M defined over

$\mathbb{AP} = \{\mathrm{isConnected}(\mathrm{pump}_1, \mathrm{ivLine}_1), \mathrm{poweredOn}(\mathrm{pump}_1), \mathrm{alarmOn}(\mathrm{pump}_1), \dots\}$

A state q_1 in the Kripke structure corresponding to

the infusion pump system might have labeling:

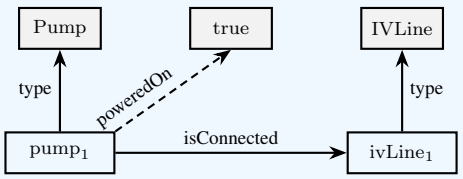
$$L(q_1) = \{\text{isConnected}(\text{pump}_1, \text{ivLine}_1), \text{poweredOn}(\text{pump}_1), \neg\text{alarmOn}(\text{pump}_1)\},$$

representing *"the pump is powered and connected to an IV line, but the alarm system is disabled."*

Reachability. For states $p, q \in Q$, we write $p \xrightarrow{*} q$ to denote that q is reachable from p. Formally, there exists a finite path $p = q_0, q_1, \ldots, q_n = q$ such that $\langle q_i, q_{i+1}\rangle \in R$ for all $0 \leq i < n$.

RDF Triples. We represent KGs using the RDF (Resource Description Framework) standard, encoding knowledge as *triples* $\langle s, p, o\rangle$ (subject, predicate, object). The type of each node is represented using a type-triple $\langle n, \text{type}, c\rangle$ for node $n \in N$ and concept $c \in \ell(n)$, while an edge becomes a triple $\langle s, p, o\rangle$ for subject node $s \in N$, predicate $p \in \ell(\langle s, o\rangle)$, and object node $o \in N$. Finally, $\mathbb{S}$, $\mathbb{P}$, and $\mathbb{O}$ denote sets of all possible subjects, predicates, and objects.

Running Example 3. Given the documentation of the infusion pump system we extract a KG as:



where solid edges represent structural relationships (types and connections) and dashed edges represent operational modes (e.g., powered on/off).

RDF Triples. The above KG can be represented as:

$$K = \{\langle \text{pump}_1, \text{type}, \text{Pump}\rangle, \langle \text{ivLine}_1, \text{type}, \text{IVLine}\rangle, \langle \text{pump}_1, \text{poweredOn}, \mathsf{true}\rangle, \langle \text{pump}_1, \text{isConnected}, \text{ivLine}_1\rangle\}.$$

representing *"the pump is powered and connected to an IV line, but the alarm system is disabled."*

$$\mathbb{S} = \{\text{pump}_1, \text{ivLine}_1, \text{battery}_1, \text{alarm}_1\}$$
$$\mathbb{P} = \{\text{alarmOn}, \text{isConnected}, \text{isInfusing}, \text{lowBattery}, \text{poweredOn}\}$$
$$\mathbb{O} = \{\text{ivLine}_1, \mathsf{true}, \mathsf{false}\}$$

The *alphabet* determines the entities and relations that can appear in the model. Under the closed-alphabet assumption, we fix finite sets of subjects $\mathbb{S}$, predicates $\mathbb{P}$, and objects $\mathbb{O}$ representing the vocabulary. These sets capture the complete vocabulary available for expressing knowledge within the Kripke structure. The alphabet assumption is orthogonal to the Closed World Assumption (CWA)/OWA distinction: alphabet closedness concerns the vocabulary, while CWA/OWA concerns truth valuations over that vocabulary. In this work, we adopt a *closed alphabet assumption*.

3. Knowledge Graph Semantics

Safety documentation often contains modal statements (e.g., *"must activate"*, *"can occur"*) that express constraints, obligations, or possibilities rather than facts. Our semantics handles these modalities by partitioning them into distinct states, each representing a valid interpretation of the documentation. We define a formal semantics for knowledge graphs using Kripke structures; this provides coverage guarantees by enumerating all possible truth valuations for undecided predicates.

Atomic Propositions and Literals. Each RDF triple $\langle s, p, o\rangle$ is converted into predicate notation $p(s, o)$ and treated as an atomic proposition.

Definition 3.1. Given a KG $K = \langle N, E, \ell\rangle$ of an ontology $\mathcal{O} = \langle \mathcal{C}, \mathcal{R}, \mathcal{A}\rangle$, we construct the set of atomic propositions from two sources: node predicates P_N (representing entity types) and edge predicates P_E (representing entity relations). These are formally defined as:

- $P_N = \{\text{type}(n, c) \mid c \in \ell(n), n \in N\}$
- $P_E = \{p(s, o) \mid p \in \ell(\langle s, o\rangle), \langle s, o\rangle \in E\}$
- $\mathbb{AP} = P_N \cup P_E$

A literal is either an atomic proposition ap or its negation $\neg ap$. We denote universal literal set by $\mathbb{L} = \mathbb{AP} \cup \{\neg p \mid p \in \mathbb{AP}\}$. Finally, $ap \in \mathbb{AP}$ is true in state q if $ap \in L(q)$, false if $\neg ap \in L(q)$, and $\mathsf{undecided}$ if neither ap nor $\neg ap$ are in $L(q)$.

Formula notation. A knowledge graph K has a formula $\varphi(K)$ over atomic propositions, and states in Kripke M satisfying this formula are models of K.

Running Example 4. Triples convert to predicates:

- $\langle \text{pump}_1, \text{type}, \text{Pump}\rangle \mapsto \text{type}(\text{pump}_1, \text{Pump})$,
- $\langle \text{pump}_1, \text{poweredOn}, \text{true}\rangle \mapsto \text{poweredOn}(\text{pump}_1)$,
- $\langle \text{pump}_1, \text{isConnected}, \text{ivLine}_1\rangle \mapsto \text{isConnected}(\text{pump}_1, \text{ivLine}_1)$.

$\mathbb{S}, \mathbb{P}, \mathbb{O}$ yield following node and edge predicates:

$$P_N = \{\text{type}(\text{pump}_1, \text{Pump}), \text{type}(\text{ivLine}_1, \text{IVLine}), \text{type}(\text{battery}_1, \text{Battery}), \text{type}(\text{alarm}_1, \text{Alarm})\},$$
$$P_E = \{\text{isConnected}(\text{pump}_1, \text{ivLine}_1), \text{poweredOn}(\text{pump}_1), \text{alarmOn}(\text{pump}_1), \text{lowBattery}(\text{pump}_1), \text{isInfusing}(\text{pump}_1)\},$$

where $\mathbb{AP} = P_N \cup P_E$. However, for brevity, in running examples, we assume $\mathbb{AP} = P_E$.

Running Example 5. Given $\mathbb{AP} = P_E$ and knowledge graph K, its formula notation is:

$$\varphi(K) = \text{poweredOn}(\text{pump}_1) \wedge \text{isConnected}(\text{pump}_1, \text{ivLine}_1).$$

3.1. *Kripke Construction*

Algorithm 1 enumerates all truth valuations over $\mathbb{AP}$ by exploring both valuations of each edge predicate in P_E, guaranteeing coverage: no feasible valuation is overlooked.

Algorithm 1 Kripke Structure Construction

Input: P_N, P_E
Output: $M = \langle Q, I, R, L\rangle$
$q_0 = \bigwedge \text{type}(\cdot)$ for $\text{type}(\cdot) \in P_N$
$Q \leftarrow \{q_0\}$, $I \leftarrow \{q_0\}$, $R \leftarrow \emptyset$
queue $\leftarrow \{q_0\}$ ▷ *queue of states to explore*
while *queue* $\neq \emptyset$ **do**
  $q \leftarrow$ pop from *queue*
  extended $\leftarrow$ **false** ▷ *is extended flag*
  for $p(\cdot) \in P_E$ where $q \not\models p(\cdot) \wedge q \not\models \neg p(\cdot)$ **do**
    $q^+ \leftarrow q \wedge p(\cdot)$, $q^- \leftarrow q \wedge \neg p(\cdot)$
    if $q^+ \notin Q$ **then**
      $Q \leftarrow Q \cup \{q^+\}$, $R \leftarrow R \cup \{\langle q, q^+\rangle\}$
      queue $\leftarrow$ *queue* $\cup \{q^+\}$, *extended* $\leftarrow$ **true**
    if $q^- \notin Q$ **then**
      $Q \leftarrow Q \cup \{q^-\}$, $R \leftarrow R \cup \{\langle q, q^-\rangle\}$
      queue $\leftarrow$ *queue* $\cup \{q^-\}$, *extended* $\leftarrow$ **true**
  if *extended* $=$ **false then**
    $R \leftarrow R \cup \{\langle q, q\rangle\}$ ▷ *stable terminal state*
Set $L(q) = q$ for all $q \in Q$
return $M = \langle Q, I, R, L\rangle$

Initial-state convention. Type predicates are fixed context. Let $T_{\text{type}} = \{\text{type}(n, c) \mid c \in \ell(n), n \in N\}$. We include T_{type} in every state labeling, so $L(q_0) = T_{\text{type}}$ and $T_{\text{type}} \subseteq L(q)$ for all $q \in Q$. Informal statements that say "all propositions are undecided" refer only to mode-dependent predicates in P_E.

Exhaustive branching. For each state q, the construction branches on every undecided predicate $p \in P_E$, creating successors q^+ with p asserted true and q^- with p asserted false, and adding transitions $q \to q^+$ and $q \to q^-$. When a state cannot be extended without contradicting axioms $\mathcal{A}$, it receives a self-loop, marking it as a terminal state. A terminal state q is one where for every $p \in P_E$, either $p \in L(q)$ or $\neg p \in L(q)$ (all propositions are decided).

Weakest state and terminal cover. For a knowledge graph K, we define the weakest state q_K as the state with the smallest labeling (minimal $|L(q)|$) that satisfies all facts in K. Formally, q_K is the state minimizing the number of decided literals while satisfying K:

$$q_K = \arg\min\{|L(q)| : q \in Q, \forall \langle s, p, o\rangle \in K : p(s, o) \in L(q)\}.$$

The terminal cover T_K is the set of terminal states reachable from q_K:

$$T_K = \{q \in Q \mid \langle q, q\rangle \in R \wedge q_K \xrightarrow{*} q\}.$$

Running Example 6. Given the K for the infusion pump system, the weakest state q_K has labeling

$$L(q_K) = \{\text{poweredOn}(\text{pump}_1), \text{isConnected}(\text{pump}_1, \text{ivLine}_1)\},$$

leaving $\text{alarmOn}(\text{pump}_1)$, $\text{lowBattery}(\text{pump}_1)$, and $\text{isInfusing}(\text{pump}_1)$ undecided; yielding, the terminal cover $T_K = \{q \in T : L(q) \models \varphi(K)\}$ contains $2^3 = 8$ feasible completions. In this example, q_K encodes $\varphi(K)$ over the mode-dependent predicates, so this satisfiability characterization coincides with the canonical reachability definition of T_K.

Axiom-driven pruning. If axioms $\mathcal{A}$ are correct and complete, they can eliminate inconsistent

states during construction without undermining coverage guarantees. However, incorrect or incomplete axioms may prune valid states and thus yield incorrect coverage guarantees. Expert validation of axioms $\mathcal{A}$ is essential prior to applying pruning.

Running Example 7. Denote all terminal states in Kripke structure M by

$$T = \{q \in Q : \langle q, q \rangle \in R\}\,.$$

With 5 Boolean predicates, we have $|T| = 32$. Axiom (A1) *"infusing requires power"* prunes 8 inconsistent states where $\text{isInfusing}(\text{pump}_1) = \mathsf{true}$, but $\text{poweredOn}(\text{pump}_1) = \mathsf{false}$, leaving $|T| = 24$ consistent terminals.

3.2. *Knowledge Graph Validation*

Given a knowledge graph K and its Kripke structure M (constructed from the alphabet), validation consists of computing the terminal cover T_K as defined above. The cardinality of T_K determines the validity of K:

- $|T_K| = 0$: K is *invalid* (violates axioms $\mathcal{A}$). The assertions of K are mutually incompatible given the axioms; the axioms prune states in M that would otherwise model K.
- $|T_K| = 1$: K is *valid and complete* (unique state). K fully specifies all predicates, yielding a unique interpretation.
- $|T_K| > 1$: K is *incomplete* (multiple consistent interpretations). K underspecifies predicates, leaving multiple consistent valuations.

Each terminal state $q_i \in T_K$ represents a distinct state satisfying both the facts of K and axioms $\mathcal{A}$. Coverage is guaranteed by construction: Algorithm 1 enumerates all feasible truth valuations, ensuring no latent hazard arising from an undecided predicate is overlooked.

3.3. *Latent Hazard Identification*

After validating a knowledge graph K (Section 3.2), we assess safety compliance by classifying terminal states in the terminal cover T_K according to safety properties. This enables systematic identification of *latent hazards*; i.e., hazards that remain hidden because the documentation is incomplete.

Hazard patterns and safety properties. We distinguish hazard patterns from safety properties. A hazard pattern is an undesirable propositional formula whose truth at a terminal state indicates a failure mode. A safety property asserts that hazard patterns do not hold. For hazard patterns $\{\phi_i\}$, we define

$$\phi_{\text{safe}} = \bigwedge_i \neg\phi_i\,.$$

Safety Properties. We classify terminal states using safety properties ϕ, expressed as propositional formulas over atomic propositions. A terminal state q is *safe* if $\bigwedge L(q) \models \phi$, and *unsafe* otherwise. These properties can be derived from OWL axioms, SWRL rules, or SHACL constraints following standard practices in semantic web reasoning (Baader, 2003; Heflin et al., 2007).

Running Example 8. Consider safety property

$$\phi_{\text{safe}} = \text{isInfusing}(\text{pump}_1) \rightarrow (\text{alarmOn}(\text{pump}_1) \wedge \text{isConnected}(\text{pump}_1, \text{ivLine}_1))\,,$$

meaning *"if infusing, the alarm must be enabled and the IV line must be connected."* A state $q \in T_K$ with labeling

$$L(q) = \{\text{isConnected}(\text{pump}_1, \text{ivLine}_1), \text{poweredOn}(\text{pump}_1), \text{isInfusing}(\text{pump}_1), \text{alarmOn}(\text{pump}_1), \ldots\}$$

satisfies this property because $L(q) \models \phi_{\text{safe}}$, while

$$L(q') = \{\text{isConnected}(\text{pump}_1, \text{ivLine}_1), \text{poweredOn}(\text{pump}_1), \text{isInfusing}(\text{pump}_1), \neg\text{alarmOn}(\text{pump}_1), \ldots\}$$

violates it because $L(q') \not\models \phi_{\text{safe}}$.

Realizability. Property ϕ is *realizable* if at least one safe terminal state is reachable in M from initial states; otherwise, ϕ is *unrealizable*.

Let a knowledge graph K denote the system documentation, with $\varphi(K)$ as its formula representation. Let T_K denote the terminal cover of K (as defined in Section 3). For each terminal state in the terminal cover, we check if its labeling satisfies safety property; i.e., $\forall q_i \in T_K : \bigwedge L(q_i) \models \phi$. The satisfaction patterns reveal distinct categories:

Complete and Consistent Documentation. If all terminal states in the terminal cover T_K satisfy the property ϕ, then the knowledge graph K is complete and consistent wrt. ϕ.

Unreachable Safety. If no terminal state in its terminal cover T_K satisfies property ϕ, then no valuation reachable from the weakest state q_K of K is safe. This indicates that either ϕ is unrealizable in Kripke M (no safe terminal state exists anywhere), or the knowledge graph K is inherently unsafe (safe terminal states exist in M but are unreachable from the weakest state q_K).

Known Unknowns. If some terminal states satisfy ϕ while others do not, then the documentation is incomplete and latent hazards remain possible; formally, $\exists q_s \in T_K : \bigwedge L(q_s) \models \phi$ and $\exists q_h \in T_K : \bigwedge L(q_h) \not\models \phi$. This indicates that the knowledge graph K is underspecified: under OWA it may not violate ϕ, but under CWA, some reachable states violate ϕ, revealing latent hazards.

Hazard Severity. For each unsafe terminal state q_h, we measure hazard severity as the minimal symmetric difference $|L(q_h) \triangle L(q_s)|$ to any safe terminal state q_s, i.e., the minimal number of literal changes required to achieve a safe labeling. A lower severity indicates closer affinity to the intent of the designer, while a higher severity indicates many predicate changes are needed, suggesting a systematic safety gap.

Running Example 9. For the medical pump example, we partition the terminal cover T_K (containing eight states) according to the safety property ϕ_{safe}. Six of the eight terminal states satisfy ϕ_{safe}. The two unsafe states are known unknowns: latent hazards in which $\text{isInfusing}(\text{pump}_1) = \mathsf{true}$ but $\text{alarmOn}(\text{pump}_1) = \mathsf{false}$.

The knowledge graph K asserts power and connection but leaves alarm status undecided, thereby permitting these unsafe configurations. This reveals latent hazards resulting from incomplete documentation. The coverage gap between OWA and CWA is $2/8 = 25\%$.

4. Automotive Seatbelt Warning System

We present a simplified automotive seatbelt system case study to demonstrate systematic identification of latent hazards. It illustrates how our methodology increases objectivity and reduces reliance on expert judgment in safety analysis.

We chose this simple, widely understood system to help readers follow the core contributions and assess the practical impact of our approach. Despite its simplicity, a seatbelt system contains subtle hazards that conventional analyses have historically overlooked. The seatbelt system is regulated worldwide (FMVSS 208, ECE R16, Euro NCAP). This simplification is a scope restriction, not an assumption that real systems are simple: we isolate one regulated function to make all modeling assumptions explicit and auditable. The same construction applies to larger systems by expanding the predicate set and partitioning the model into interacting subsystems.

Fastened Seatbelt. The system includes a seatbelt with a buckle. The buckle can be buckled (latched) but not fastened (securely restraining the occupant). This distinction is critical for safety: a buckled but unfastened seatbelt may fail to restrain the occupant during a crash, while a fastened seatbelt provides proper restraint. However, documentation often conflates these states, leading to latent hazards where the system may incorrectly assume safety when the seatbelt is only buckled but not fastened.

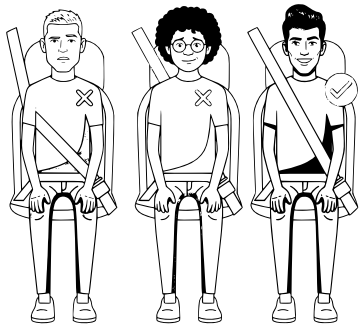


Fig. 1. A fastened seatbelt restrains the upper body.

Knowledge Graph. We extract a KG from system documentation and observations. Following are the concepts $\mathcal{C}$, relations $\mathcal{R}$, and nodes N.

$$\begin{aligned}\mathcal{C} &= \{\text{Ignition}, \text{Seat}, \text{Seatbelt}, \text{SeatbeltBuckle},\\ &\quad \text{Alarm}, \text{Vehicle}, \text{Person}\}\\ \mathcal{R} &= \{\text{has}, \text{isActive}, \text{isBuckled}, \text{isFastened}, \text{isOn},\\ &\quad \text{isOccupied}, \text{speedExceeds}, \text{speedLimit}\}\\ N &= \{\text{alarm}, \text{driver}, \text{driverSeat}, \text{ignition},\\ &\quad \text{seatbelt}, \text{seatbeltBuckle}, \text{vehicle}\}\end{aligned}$$

The documentation for the system specifies:

> *The alarm activates if and only if the ignition is on, the seat is occupied, and the seatbelt is not buckled. When these conditions hold, the speed limiter must engage.*

The formula $\varphi(K)$ of the knowledge graph K is:

$$\begin{aligned}\varphi(K) =&[\text{isActive} \iff (\text{isOn} \wedge\\ &\quad \text{isOccupied} \wedge \neg\text{isBuckled})]\\ &\wedge[(\text{isOn} \wedge \text{isOccupied} \wedge \neg\text{isBuckled})\\ &\quad \implies \text{speedLimit}]\,.\end{aligned}$$

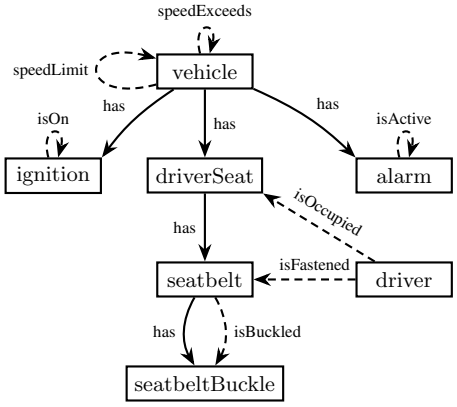


Fig. 2. Simplified seatbelt system (for one seat).

Figure 2 shows the extracted knowledge graph of the simplified seatbelt system. Ontological type relationships (e.g., driverSeat is a Seat) are omitted for brevity. Solid arrows represent structural relations (has) and dashed arrows represent mode-dependent relations that can vary across operational modes (e.g., speedExceeds).

Kripke Structure. Node predicates P_N are fixed across states. Edge predicates P_E include both structural relations (e.g., has) and mode-dependent relations. We focus on seven binary mode-dependent predicates:

$$\begin{aligned}&\text{isOn}(\text{ignition},\ \text{true})\,,\\ &\text{isOccupied}(\text{driverSeat},\ \text{true})\,,\\ &\text{isFastened}(\text{driver},\ \text{seatbelt})\,,\\ &\text{isBuckled}(\text{seatbelt},\ \text{seatbeltBuckle})\,,\\ &\text{isActive}(\text{alarm},\ \text{true})\,,\\ &\text{speedExceeds}(\text{vehicle},\ 20\ \text{km/h})\,,\\ &\text{speedLimit}(\text{vehicle},\ 20\ \text{km/h})\end{aligned}$$

For brevity, we omit explicit arguments of predicates and refer to $\text{isFastened}(\text{driver}, \text{seatbelt})$ simply as isFastened, and so forth.

Considerations. We distinguish sensor observations (e.g., speedExceeds) from actuator outputs (e.g., speedLimit). We assume structural completeness: the seatbelt-warning subsystem contains all structural components relevant to the analyzed predicates. While our methodology supports verifying structural completeness, here we only focus on validating and verifying behavioral completeness and correctness, which present greater analytical challenges and require substantial expert effort.

Ontological Axioms. Four axioms constrain the state space. The core axiom is:

A1. *If the person is properly wearing the seatbelt for restraint, then the buckle must be latched.*

$$\text{isFastened} \implies \text{isBuckled}\,.$$

Three additional axioms are optionally enforced depending on system assumptions:

A2. *Belt cannot be fastened with no occupant:*

$$\neg\text{isOccupied} \implies \neg\text{isFastened}\,.$$

A3. *Buckle cannot be latched with no occupant:*

$$\neg\text{isOccupied} \implies \neg\text{isBuckled}\,.$$

A4. *With the ignition off, speed cannot exceed 20 km/h:*

$$\neg\text{isOn} \implies \neg\text{speedExceeds}\,.$$

Table 1 shows axiom-driven pruning. With 7 binary mode-dependent predicates, the theoretical maximum is 128 terminals. Each axiom combination prunes infeasible states, with the number of unsafe terminals varying depending on which operational contexts are deemed feasible.

4.1. *Latent Hazard Detection Framework*

To identify latent hazards, we compute the terminal cover T_K of the extracted KG using the canonical reachability definition from Section 3. When q_K encodes $\varphi(K)$ on the chosen alphabet, this is equivalent to the satisfiability characterization $T_K = \{q \in T : L(q) \models \varphi(K)\}$. Notably, the documented system is assumed to be safe.

Hazard Patterns. We specify two hazard patterns capturing latent hazards:

$$\phi_{\text{fasten}} = \text{isBuckled} \wedge \neg\text{isFastened}$$
$$\phi_{\text{speed}} = \text{speedLimit} \wedge \text{speedExceeds}$$

and define the corresponding safety property as

$$\phi_{\text{safe}} = \neg\phi_{\text{fasten}} \wedge \neg\phi_{\text{speed}}\,.$$

We classify states into three tiers: *documented safety* (terminal states that satisfy $\varphi(K)$), *misuse hazard* (terminal states that satisfy the hazard pattern ϕ_{fasten}), and *enforcement hazard* (terminal states that satisfy the hazard pattern ϕ_{speed}). A state is *truly safe* iff it satisfies $\varphi(K)$ and ϕ_{safe}.

Table 1. State space size by axiom-driven pruning.

#	Axioms	$\lvert Q\rvert$	Safe	Unsafe	Pruned
1	None	128	60	68	0
2	A1	96	46	50	32
3	A2	96	44	52	32
4	A3	96	44	52	32
5	A4	96	44	52	32
6	A1+A2	80	38	42	48
7	A1+A3	64	30	34	64
8	A1+A4	72	34	38	56
9	All (A1–A4)	48	22	26	80

Table 2. Latent hazard analysis with all axioms enabled.

Classification	Count	% of Safe
States satisfying $\varphi(K)$	22	100%
Truly safe (no latent hazards)	13	59.1%
Misuse hazard (ϕ_{fasten})	5	22.7%
Enforcement hazard (ϕ_{speed})	3	13.6%
Both latent hazards	1	4.5%
Coverage gap	**9**	**40.9%**

With all four axioms enabled (A1–A4), we obtain 48 terminal states: 22 satisfy $\varphi(K)$ and 26 are unsafe. Table 2 shows that fewer than 60% of documented-safe states are truly safe, with the remainder harboring latent hazards. We now examine each hazard type in detail.

4.2. *Misuse Hazards*

Documentation conflates person restrained (i.e., isFastened) and buckle sensor triggered (i.e., isBuckled), implicitly assuming:

$$\text{isFastened} \iff \text{isBuckled}\,.$$

Realistically, designs only enforce:

$$\text{isFastened} \implies \text{isBuckled}\,.$$

The reverse fails: occupants can buckle behind their back or under their arm without proper restraint. This creates a misuse hazard where the system observes $\text{isBuckled} = \mathsf{true}$ but the actual safety state is $\text{isFastened} = \mathsf{false}$.

Five states satisfy $\varphi(K)$ but satisfy ϕ_{fasten}. The system believes these states are safe (buckle latched, alarm off, speed unrestricted), yet the occupant lacks crash protection. The $\varphi(K)$ uses only isBuckled because documentation can only reference measurable system states, but this leaves the misuse case undetected.

4.3. *Enforcement Hazards*

Documentation states *"speed is limited to 20 km/h when unfastened"* without distinguishing the control action (i.e., speedLimit) from the observation (i.e., speedExceeds). While $\varphi(K)$ correctly uses speedLimit, the specification assumes $\text{speedExceeds} = \mathsf{false}$ if $\text{speedLimit} = \mathsf{true}$.

In low-speed scenarios (15 km/h with unbuckled belt), engaging the speed limiter safely prevents acceleration beyond 20 km/h. However, in high-speed scenarios (120 km/h on highway when belt becomes unbuckled), the same control action creates a critical hazard: abrupt deceleration

causes loss of control and multi-vehicle collisions (Hu et al., 2016; Soica and Gheorghe, 2025).

Three states satisfy $\varphi(K)$ but satisfy ϕ_{speed}: the speed limiter is engaged ($\text{speedLimit} = \text{true}$) while the vehicle travels at high speed ($\text{speedExceeds} = \text{true}$). This represents a *temporal hazard*: the state denotes a dangerous transient in which active enforcement causes abrupt deceleration. Although speedExceeds should eventually become false, the path to compliance is hazardous. Our methodology exposes this by treating specifications as properties to verify, not requirements to implement (Rushby, 2009).

4.4. *Summary and Implications*

Beyond the two individual hazard types, one state exhibits both misuse and enforcement hazards simultaneously: belt buckled at high speed and the system attempts deceleration enforcing safe speed limits. This compounds risks, as the occupant is unrestrained during dangerous braking.

This coverage gap shows that nearly half of specification-compliant states harbor undocumented hazards. Traditional validation that checks only $\varphi(K)$ would approve these states and thereby miss latent hazards arising from conflating observables with unobservables or control outputs with sensor inputs. Our exhaustive enumeration provides formal assurance that all hazards within the modeled scope are identified.

5. Related Work

OWA-based reasoners check consistency but do not necessarily identify latent hazards in incomplete specifications, nor do they distinguish false negatives (known unknowns) from false positives (contextual ambiguities) (Baader, 2003; Heflin et al., 2007; Knublauch and Kontokostas, 2017). Gueffaz et al. (2012) translate RDF to Kripke structures by treating states as nodes for navigation, not as truth valuations. They neither branch to materialize known unknowns, nor expose contextual ambiguities.

Hazard analysis rely on checklists and expert judgment, making it fundamentally incomplete and error-prone as they depend on analysts imagining scenarios (Leveson, 2012). We mechanized risk identification by enumerating possible valuations and transforming previously implicit latent hazards into explicit findings with guarantees.

Temporal RDF extensions address temporal modalities but generally lack dedicated safety-analysis capabilities (Hartig and Thompson, 2014). Our contribution bridges ontological knowledge graphs with modal CWA reasoning, leveraging mature formal-methods tooling while preserving the flexibility of OWA. Model checking verifies properties over complete models under the closed-world assumption (Clarke et al., 2018). This work complements CWA-based verification by extending it to incomplete models under OWA via construction-time reasoning.

Embedding-based KG completion (Ji et al., 2022) predicts missing triples statistically but cannot distinguish documented from latent safety requirements. Our contribution explicitly models this distinction through three-tier classification (documented safety, misuse hazards, enforcement hazards), enabling coverage gap quantification that embedding methods cannot provide. Statistical approaches optimize for prediction accuracy on historical data; our exhaustive enumeration guarantees coverage over all feasible states within the declared alphabet.

6. Conclusion

We present Kripke semantics with exhaustive branching that guarantee coverage over all truth valuations, together with three identification patterns that discover latent hazards (known unknowns). Construction-time validation treats contradictory documentation as distinct states rather than as errors that require prior resolution, thereby reducing reliance on expert judgment.

The case study demonstrates that the gap between OWA and CWA manifests in real systems: documentation exhibits critical conflations. First, it conflates predicates like isFastened and isBuckled that appear synonymous but carry different safety implications (observable sensor state vs. unobservable proper restraint). Second, it conflates speedLimit (control output) and speedExceeds (sensor input), creating enforcement hazards where engaging the speed limiter at highway

speeds causes dangerous abrupt deceleration. Our methodology exposed these hazards through exhaustive enumeration: with 7 predicates generating 128 possible states (48 feasible under axioms), we identified 22 safe and 26 unsafe states. Crucially, 40.9% of states marked safe by documented specifications harbor latent hazards, demonstrating systematic underspecification.

Our framework distinguishes *documented safety* (satisfaction of explicit requirements) from *actual safety* (absence of all hazards including latent ones). This distinction is critical for:

- **Accountability:** When hazards manifest, was the state documented-safe but actually-unsafe (specification gap) or documented-unsafe (implementation failure)?
- **Completeness:** Coverage gap quantifies specification quality
- **Traceability:** Latent hazards trace to missing documentation, not faulty implementation

By exposing known unknowns as explicit terminals, the framework enables precise attribution when hazards occur: the Kripke structure reveals whether risks were enumerated and accepted, or genuinely unforeseen. This distinction matters for accountability where liability depends on demonstrating due diligence. Identifying contextual ambiguities prevents engineers from implementing specifications that are safe in one context but hazardous in another, which is a class of errors that has caused incidents across various domains.

As with many formal verification methods, our approach has exponential worst-case time and memory complexity in the number of mode-dependent predicates (exponent $|P_E|$). This is the cost of exhaustive coverage. In practice, we mitigate complexity via compositional decomposition, symbolic encodings (SAT/SMT or BDD), and property-guided reductions; full scalability evaluation is deferred to future work.

Concerning external validity, the seatbelt study is a controlled proof-of-concept illustrating the failure modes that the method detects. Application to real-world systems is feasible when documentation is structured into requirements, hazards, and trace links; in that setting, the method scales through modular analysis and staged integration rather than monolithic modeling.

Extending this work requires several advances: methods to address unknown unknowns through iterative ontology refinement beyond closed-alphabet assumptions; scalability techniques (partial order reduction, symbolic encodings, incremental validation); and automated repair strategies (for example, delta-debugging approaches (Zeller, 1999; Ebrahimi et al., 2026)) to convert unsafe terminals into safe ones.

Acknowledgement

This work was supported by the ASSURE project (MDU multidisciplinary research initiative) and in part by the ITEA4 24026 CLEAR (Comprehensive Learning for Enhanced AI Responsiveness) project.

References

Alur, R. (2015). *Principles of Cyber-Physical Systems*. Cambridge, MA, USA: The MIT Press.

Baader, F. (2003). *The Description Logic Handbook: Theory, Implementation, and Applications*. New York, NY, USA: Cambridge University Press.

Clarke, E. M., O. Grumberg, D. Kroening, D. A. Peled, and H. Veith (2018). *Model Checking* (2nd ed.). Cambridge, MA, USA: MIT Press.

Ebrahimi, M., K. Hänninen, K. Lundqvist, and M. Sirjani (2026). When Repair is not Enough: Systematic Mitigation Strategies for Incomplete Knowledge Graphs. Submitted to ESREL 2026. Companion paper presenting specification-level mitigation framework.

Ehrlinger, L. and W. Wöß (2016). Towards a definition of knowledge graphs. *SEMANTiCS 48*(1-4), 2.

Gruber, T. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition 5*(2), 199–220.

Gueffaz, M., S. Rampacek, and C. Nicolle (2012). Temporal logic to query semantic graphs using the model checking method. *J. Softw. 7*(7), 1462–1472.

Hall, A. T., D. D. Frink, and M. R. Buckley (2017). An accountability account: A review and synthesis of the theoretical and empirical research on felt accountability. *Journal of Organizational Behavior 38*(2), 204–224.

Hartig, O. and B. Thompson (2014). Foundations of an alternative approach to reification in rdf. *CoRR abs/1406.3399*.

Heflin, J. et al. (2007). An introduction to the owl web ontology language. *Lehigh University. National Science Foundation (NSF) 7*.

Hu, J., J. D. Rupp, K. Fischer, P. Lange, and A. Adler (2016). Optimizing seat belt and airbag designs for rear seat occupant protection in frontal crashes. In *Proceedings of the IRCOBI Conference 2016*, pp. 67–100.

ISO (2018). BS EN ISO 9241-304:2008: Ergonomics of human-system interaction. ISO standard.

Ji, S., S. Pan, E. Cambria, P. Marttinen, and P. S. Yu (2022). A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Transactions on Neural Networks and Learning Systems 33*(2), 494–514.

Knublauch, H. and D. Kontokostas (2017, July). Shapes constraint language (shacl). W3c recommendation, W3C.

Leveson, N. G. (2012). *Engineering a Safer World: Systems Thinking Applied to Safety*. MIT Press.

Musen, M. A. (2015). The protégé project: a look back and a look forward. In *AI Matters*, Volume 1, pp. 4–12. ACM.

Palshikar, G. K. (2001). Applying formal specifications to real-world software development. *IEEE Softw. 18*(6), 89–97.

Rushby, J. (2009). Software verification and system assurance. In *SEFM 2009*, pp. 3–10. IEEE Computer Society.

Soica, A. and C. Gheorghe (2025). A review of seatbelt technologies and their role in vehicle safety. *Applied Sciences 15*(10).

Zeller, A. (1999). Yesterday, my program worked. today, it does not. why? In *Software Engineering — ESEC/FSE '99*, Volume 1687 of *Lecture Notes in Computer Science*, pp. 253–267. Springer.