Formal Modeling of Trust in AI-Driven Autonomous Delivery Vehicles

Manar Altamimi[1], Asieh Salehi Fathabadi[2], Vahid Yazdanpanah[2]

[1]School of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Saudi Arabia.
[2]School of Electronics and Computer Science, University of Southampton, UK.

Contributing authors: mmaltamimi@pnu.edu.sa; a.salehi-fathabadi@soton.ac.uk; v.yazdanpanah@soton.ac.uk;

Abstract

Trust modeling is critical for the safe deployment of autonomous systems, yet existing approaches that rely primarily on historical performance data fail to capture dynamic operational contexts and real-time agent capabilities. This paper introduces a formal framework for modeling actual trust in Autonomous Delivery Vehicles (ADVs)—a context-aware trust model that evaluates an agent's current ability, knowledge state, and commitment to task completion rather than relying solely on past behavior. We present a systematic refinement-based approach using Event-B formal methods to model trust in ADV task delegation scenarios. Our methodology progresses through five refinement levels, transitioning from an untrusted baseline model to a comprehensive trust framework that integrates three key dimensions: (1) strategic trust (capability verification), (2) epistemic trust (knowledge-based assessment), and (3) commitment trust (availability and willingness evaluation). Each refinement level addresses specific failure modes identified in traditional delegation systems where tasks may be assigned to incapable, unknown, or unavailable vehicles. The formal model is verified using the Rodin theorem prover with 93 proof obligations, achieving 90% automatic verification. Our approach demonstrates how actual trust can be systematically integrated into autonomous systems through correctness-by-construction refinement, ensuring that task assignments occur only when trust conditions are formally verified. The framework provides a foundation for trustworthy task delegation in multi-agent autonomous systems and offers insights for developing reliable AI-driven delivery networks.

Keywords: Autonomous Systems, Formal Methods, trust, Event-B, trustworthy AI

1 Introduction

Trust in autonomous systems is a multifaceted and context-dependent concept that remains difficult to formalize, interpret, and verify [1, 2]. Within Multi-Agent Systems (MAS), trust plays a crucial role in enabling effective coordination and task delegation [3]. Early research in MAS, such as Ramchurn et al. [4], emphasized the complexity of trust modeling, typically relying on agents' historical behavior to infer trustworthiness. [5, 6] continue to focus on retrospective reasoning, maintaining the assumption that trust can be adequately captured through analysis of past interactions. However, traditional approaches that rely solely on past interactions are often inadequate in real-time and dynamic environments, where agents must make decisions based on their current beliefs, situational awareness, and evolving goals.

Addressing this limitation, Salehi et al. [7] introduced the concept of *actual trust* and proposed a formal Event-B [8] model that accounts for an agent's knowledge, capability, and commitment at the time of delegation. This interaction is dynamic, focusing on causality [9] rather than static.

While Salehi et al. provide a strong theoretical foundation, this paper extends the formal model of actual trust by grounding it in a practical, real-world case study: Autonomous Delivery Vehicles (ADVs). ADVs are a compelling domain due to their reliance on real-time decision-making, decentralized control, and varying operational conditions. The use of formal methods to verify trust-related behavior in such systems is crucial for ensuring safety, reliability, and accountability.

To illustrate the limitations of traditional trust models, consider a scenario in which a delivery agent must delegate a package delivery to one of several autonomous vehicles. A traditional trust model might select a vehicle based on past successful deliveries. However, this could result in assigning the task to a vehicle that is currently unavailable, malfunctioning, or unaware of its own limitations. This mismatch can lead to delivery failure despite the vehicle's favorable history. In contrast, a trust model that considers current capability, current knowledge, and explicit task commitment is essential to make reliable delegation decisions.

Such limitations motivate the following research question, which builds on the theoretical foundation established in [7]:

- How can refinement-based formal methods ensure that trust assumptions remain consistent across evolving system states and interactions?

To address this research question, we extend the model of actual trust introduced in [7] by applying it to a real-world case study involving Autonomous Delivery Vehicles (ADVs), enabling a contextualized and operational evaluation of trust dimensions. This is achieved by formally developing a refinement-based model of actual trust tailored to ADVs, demonstrating how trust reasoning—grounded in dimensions such as capability, knowledge, and commitment—can be preserved in dynamic and decentralized contexts. Through a systematic refinement strategy in Event-B, the model captures the progression from abstract task delegation to trust-informed decision-making. Furthermore, the use of theorem proving within the Rodin platform ensures the consistency of trust invariants across refinements, supporting both automatic proof discharge and manual validation for complex proof obligations. The resulting model

provides a rigorous and formally verified foundation for building trust in autonomous systems that operate under uncertainty.

The paper is structured as follows: Section 2 presents background knowledge on the applied modelling approach and outlines the refinement-based strategy used to incorporate actual trust into the ADV system and background of the Event-B formal method. Section 3 describes the abstract model, establishing the foundational representation of tasks and agent interactions in an untrusted cases. Section 4 elaborates on the series of refinements that incrementally introduce and verify trust dimensions—capability, epistemic knowledge, and commitment—along with ADV state transitions. Section 5 summarizes the verification process, including the theorem proving and proof obligation analysis conducted using the Rodin platform. Section 6 discusses related work and reflects on how our model advances the current state of research in trust modeling. Finally, Section 7 concludes the paper and outlines future directions.

2 Background and Overview of Approach

This section provides the necessary background on modeling trust in autonomous systems and introduces the formal methods employed in this work. In particular, we adopt a case study of Autonomous Delivery Vehicles (ADVs) to investigate the formalization of actual trust, building upon the refinement-based modeling framework [8]. The model is specified using Event-B and is formally verified using the Rodin tools in conjunction with the iUML-B toolset [10].

2.1 Modeling Approach: Modeling Actual Trust in Autonomous Systems

A correct-by-construction refinement strategy [11, 12] has been adopted to formally model the notions of trust using the Event-B method. This approach enables the system to be developed incrementally across six successive refinements, with each refinement introducing a specific layer of system behavior. The modeling process starts with an abstract representation, in which all Autonomous Delivery Vehicles (ADVs) are considered untrusted, and gradually integrates trust-related notions to reflect a trust-based delivery scenario. The refinements are constructed and visualized using iUML-B [10], a graphical front-end to Event-B that supports the use of state machines.

The actual trust notion [7, 13] focuses on examining the *current* state in autonomous systems, specifically their ability to successfully complete a specific task within the specified time frame. Actual trust is defined as a relational notion between two agents, `i` (as the trustor agent) and `j` (as the trustee agent) and say, `i` trusts `j` with respect to task `t` only if `i` is able to verify that `j` is able and committed to deliver `t`.

To develop actual trust in ADVs, we follow a refinement-based approach. We begin with an abstract model representing the untrusted state of the system —capturing **task** , **agent capability**, and **ADV availability**—as illustrated in Figure 2.1, where no trust assumptions are yet enforced. This abstract model represents a baseline model in which tasks can be assigned to any ADV, regardless of its current capabilities, availability, or commitment to perform the given task. We then progressively integrate

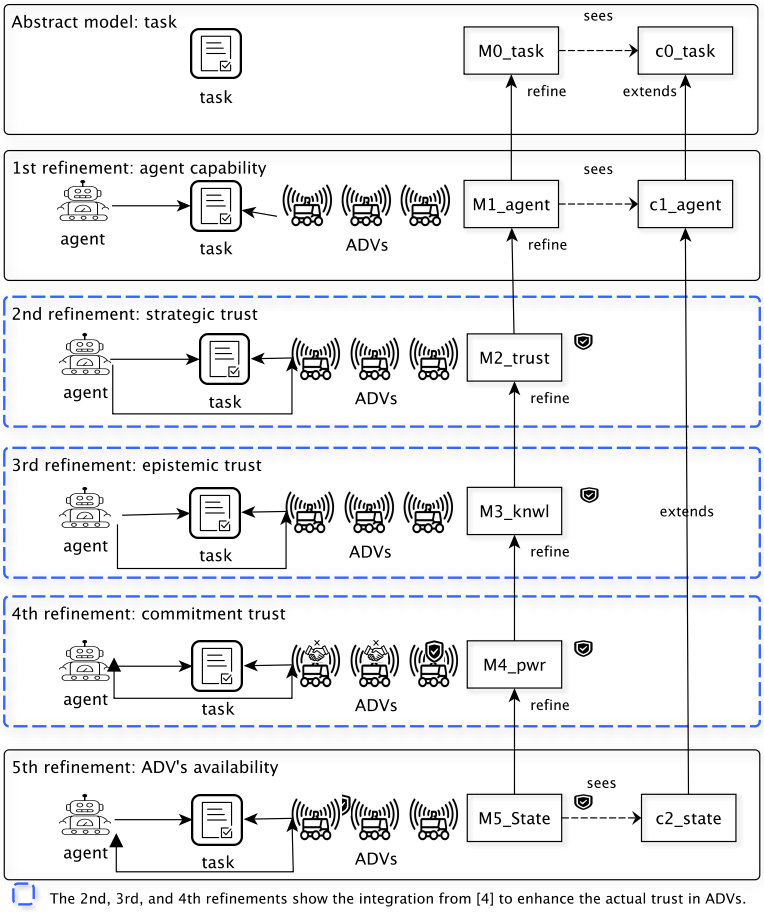

Fig. 1 Refinement Strategy to model actual trust in ADVs

the dimensions of actual trust notions, epistemic, and commitment trust, across three levels of concrete refinement, demonstrating how each level incrementally strengthens the model.

Figure 2.1 consolidates the refinement hierarchy, showing the model's progression from abstract task delegation to a fully trust-integrated ADV system. Details of these refinements are presented in Section 4.

- Strategic trust (2nd refinement): ADV capability verification
- Epistemic trust (3rd refinement): Agent's knowledge of capabilities
- Commitment trust (4th refinement): ADV availability and willingness

2.2 Event-B Formal Method

Event-B [8, 14] is a formal method for system development, particularly suitable for safety-critical and reliable systems. It supports rigorous specification and refinement through mathematical reasoning. An Event-B model is composed of two core components:

- Contexts: which define the static aspects of the system, such as carrier sets, constants, and axioms that constrain them.
- Machines: which capture dynamic behavior through variables, invariants, and events. Events describe state transitions using guards (preconditions) and actions (state updates).

Modeling and verification in Event-B is supported by the Rodin platform [8, 14], an open-source toolset built on the Eclipse framework. Rodin provides:

- Automated and interactive proof support, ensuring model consistency, invariant preservation, and refinement correctness through proof obligations.
- Model validation tools, including model checking and animation, to verify that the system behaves as expected with respect to the specified properties.

To manage the complexity of modeling actual trust in autonomous systems, especially between multiple refinement layers, we adopt iUML-B [15–17], a diagrammatic modeling notation integrated with Event-B, designed to enhance the visual expressiveness of formal models. It provides two key diagram types:

- State machines, to capture system behavior through states and transitions;
- Class diagrams, to structure data and define relationships between components.

iUML-B automatically generates corresponding Event-B elements from these diagrams, maintaining semantic consistency while improving readability and development efficiency. Its graphical interface complements the textual nature of Event-B, making the refinement process more accessible and intuitive.

Event-B formal modeling, along with the associated tools Rodin and iUML-B, supports a correct-by-construction refinement strategy and manages complexity through abstraction, making it suitable for modeling trust reasoning in autonomous systems.

3 Modelling Abstract ADVs

In a basic autonomous delivery system, tasks are dispatched to available vehicles based solely on accessibility, with no evaluation of suitability or operational readiness. The agent responsible for delegation assumes that any ADV can perform any task, regardless of its current state in terms of capability limitations, awareness of its ability, or commitment to achieving a given task.

This scenario represents an untrusted task delegation setting, in which assignments occur without validating whether the ADV is capable of completing the task, known to be reliable, or presently available and committed. The absence of such trust assessments makes the system vulnerable to task failures, inconsistent performance, and

reduced operational dependability. The primary components involved in this setting are as follows:

Tasks Each task is defined by a set of specifications, such as delivery destination, time constraints, and required capabilities (e.g., payload capacity, route range). These specifications implicitly establish a minimum threshold that the executing ADV must meet to ensure successful delivery.

Agent The agent acts as the central coordinator, maintaining a list of pending tasks and assigning them to ADVs based on accessibility alone. It lacks visibility into the internal states of the ADVs and does not evaluate whether a selected vehicle meets the task's requirements.

Autonomous Delivery Vehicle (ADV) Each ADV is assumed to be capable of executing any given task and is expected to proceed with the delivery immediately upon assignment. The vehicle is also expected to return to its origin after task completion and be ready for subsequent assignments, without the system validating its operational status or readiness.

Operating under these assumptions, the system frequently encounters task execution failures. An ADV may be assigned a delivery it cannot complete due to unmet capability requirements (e.g., insufficient payload capacity or battery range), lack of real-time availability (e.g., already engaged in another task), or internal conditions not visible to the agent (e.g., navigation errors or degraded performance). Because the agent neither verifies the ADV's suitability nor confirms its willingness or readiness to perform the task, assignments may lead to delays, incomplete deliveries, or the need for manual intervention. These outcomes not only reduce the system's efficiency but also undermine its reliability and scalability in real-case scenario.

To formally capture this failure behavior, we develop an initial untrusted Event-B model of the ADV system, in which task assignments proceed without enforcing trust notions, Figure **??**. This abstract model is composed of three levels that reflect the potential for task failure due to the absence of strategic, epistemic, and commitment guarantees, detailed in Section 4. These abstract levels are:

The abstract refinement: task defines the foundational structure of the ADV system at a high level. The *context* introduces the static elements:

- `SETS`: $TASKS$ These represent the set of deliverable tasks where autonomous delivery vehicles (ADVs) that carry them.

The *machine* defines the dynamic behavior of the system using:

- `VARIABLES`:$taskComp \in tasks \rightarrow BOOL$, $taskStart \in tasks \rightarrow BOOL$, where $taskComp$ indicates whether a task has been completed, and $taskStart$ indicates whether a task has been started.
- `INVARIANT`: $\forall t \cdot t \in tasks \land t \in dom(taskComp) \land t \in dom(taskStart) \land taskComp(t) = TRUE \Rightarrow taskStart(t) = TRUE$ ensures that a task cannot be completed unless it has already been started.

- EVENTS: define_task introduces a new task to the system with its associated specifications, start_task transitions a task into execution, and complete_task marks the task as completed and frees the ADV.

The 1st refinement: agent's capability introduces the ADV and agent sets and refines the behavior of task delegation.

The *context* introduces additional sets:

- SETS: AGENTS, ADVS
  These represent the agents responsible for task delegation and the autonomous delivery vehicles (ADVs) that perform deliveries.

The *machine* introduces:

- VARIABLES: $assignedTask \in advs \pfun tasks$, $capabilities \in advs \rel CAPABILITY$, $requiresMin \in tasks \rel CAPABILITY$, where CAPABILITY is a set of capabilities.
- EVENTS: In addition to the refined events from the previous refinement, new events are introduced such as add_adv_Capability, which refines the task assignment logic to include capability checks, and assign_advs_task to delegate tasks to capable ADVs.

The 2nd refinement: agent's states extends the system to account for the operational state of each autonomous delivery vehicle (ADV). It introduces a dynamic view of ADVs by distinguishing between different phases in their task lifecycle, such as being *available*, *ready*, *delivering*, or *returning*. This refinement allows the model to reason about the ADV's availability and readiness before task delegation.

In a subsequent section, we progressively introduce these trust dimensions to support safer and more reliable task delegation.

4 Modelling Actual Trust in ADVs

To overcome the limitations of the untrusted model described earlier in Section 3, we integrate the formal notion of trust proposed in [7], which defines trust as a relation grounded in an agent's assessment of an ADV's capability, knowledge, and commitment to perform a given task. Guided by this definition, our refinement strategy, illustrated in Figure 2.1, introduces three key dimensions of actual trust: **strategic trust** (trust in the ADV's capabilities), **epistemic trust** (trust based on the agent's knowledge of those capabilities), and **commitment trust** (trust in the ADV's readiness and willingness to execute the task).

Each refinement level addresses one of the failure scenarios identified in the abstract model in Section 3 and incrementally constrains the conditions under which task delegation can occur. This refinement development process ensures that the model evolves correctly by construction, with each level preserving the soundness and consistency of the previous one. The formal development is structured through these successive refinements, each capturing a distinct dimension of actual trust.

The integration of these trust dimensions into the formal model is achieved through a combination of carefully defined invariants and event refinement. Each dimension; strategic, epistemic, and commitment trust; is involved as a set of invariants that are

proved to be preserved across all refinement levels. Additional invariants are introduced to capture other essential properties of the ADV system, ensuring consistency and firm throughout the development.

Among the events in the model in the abstract refinements, assign_task plays a central role. It holds the assignment of a task to an ADV and acts as the essential point for enforcing trust conditions. However, assign_task is not work in isolation: it depends on the successful occurrence of prior events such as define_task, which establishes the task and its requirements, and trust, which reflects the trustor's decision to rely on a given ADV.

As trust dimensions are introduced through successive refinements, the assign_task, and trust events are incrementally strengthened to reflect the evolving trust notions. They ensure that trust-related invariants are preserved and that task delegation only occurs under verified conditions. The last refinement of this event is presented in the following and will be explained throughout the remainder of this section[1].

EVENT assign_task
ANY i j t
WHERE
@grd01 : j ∈ advs // j is the trustee ADV (M1)
@grd02 : t ∈ tasks // t is a defined task (M1)
@grd03 : capabilities[{j}] ∩ requires[{t}] ≠ ∅ // ADV j is capable of t (M1)
@grd04 : taskComp(t) = FALSE // t has not been completed (M1)
@grd05 : adv(j) = TRUST // ADV j is trusted (M2)
@grd06 : j ∈ dom(adv) // ADV j has a defined state (M2)
@grd07 : j ↦ t ∈ ran(trustor_trustee_task) // j is trusted by some agent for task t (M2)
@grd08 : adv_states(j) = available // ADV j is currently available (M4/M5)
THEN
@act01 : assignedTask(j) := t // Assign t to ADV j (M1)
@act02 : adv_states(j) := ready // Update ADV state to 'ready' (M4/M5)
END

4.1 2nd Refinement: Strategic Trust

The second refinement follows the first refinement on agent capability. It introduces the notion of strategic trust, which ensures that an ADV is only assigned a task if it possesses the necessary capabilities to perform it. We assume that all ADVs are initially in an untrusted state, and only transition to the trusted state once they satisfy the conditions for all three trust dimensions, illustred in Figure 2. This addresses the first failure scenario identified in the abstract model, where tasks may be assigned to any available ADV regardless of its suitability. In such cases, the assignment may fail because the selected ADV is unfit to complete the given task.

@inv01 : ∀ i , j · i ∈ agents ∧ j ∈ advs ∧ i ∈ dom(trustor_trustee_task) ⇒ i ≠ j
@inv02 : ∀ i , t · t ∈ tasks ∧ i ∈ agents ⇒ requiresMin[{t}] ⊆ capabilities[advs]
@inv03 : dom(assignedTask) ⊆ dom(trustees_task)

[1] The full model can be accessed here: https://shorturl.at/NiIkb

```
EVENT trust
ANY i j t
WHERE
 @grd01: j ∈ advs                          // j is the trustee ADV (M2)
 @grd02: i ∈ agents                        // i is the trustor agent (M2)
 @grd03: t ∈ tasks                         // t is a task requested by i (M2)
 @grd04: t ∈ trustees_task[{j}]            // ADV j is assigned task t (M2)
 @grd05: i ≠ j                             // An agent cannot trust itself (M2)
 @grd06: i ∉ dom(trustor_trustee_task)     // i hasn't trusted any ADV yet (M2)
 @grd07: adv(j) = NOT_TRUST                // ADV j is not yet trusted (M2)
 @grd08: j ∈ knowledge[{i}]                // Agent i knows ADV j (M3)
 @grd09: adv_untrusted(j) = committed      // ADV j has declared commitment (M4)
 @grd10: (i ↦ (j ↦ t)) ∈ dom(commitments)  // Commitment exists (M4)
 @grd11: commitments(i ↦ (j ↦ t)) = TRUE   // Commitment is confirmed (M4)
THEN
 @act01: adv(j) := TRUST                   // Set ADV j to trusted (M2)
 @act02: trustor_trustee_task :=
     trustor_trustee_task ∪ {i ↦ (j ↦ t)}  // Register trust relationship (M2)
 @act03: adv_untrusted(j) := adv_untrusted_NULL  // Reset untrusted state (M4)
 @act04: adv_states(j) := available        // ADV becomes available to act (M4)
END
```

To formally capture strategic trust M2, we introduce invariant @inv01, @inv02, and @inv03 that constrains task assignment to only those ADVs capable of fulfilling the task requirements. Each task is associated with a capability requirement, and each ADV is modeled with its own set of capabilities. Invariant inv01 ensures that a trustor agent does not assign a task to itself as a trustee ADV, maintaining the separation of delegation roles. Invariant inv02 ensures that every task required by an agent is matched against the set of capabilities possessed by available ADVs, ensuring capability feasibility across all tasks. Lastly, Invariant inv03 restricts the domain of assigned tasks to those ADVs that are already recognized as trustees, thereby ensuring that task assignments are only made to trusted and validated ADVs. These invariants collectively preserve trust correctness even as ADVs transition between operational states. The assign_task event is refined at this level by incorporating additional guards @grd05,@grd06, @grd07 which enforces the strategic trust notions.

Specifically, this guard @grd05 checks that the ADV j is in a trusted state, which introduced the trust variable adv. This means j has already been evaluated its capability and has been formally marked as TRUSTED. This is necessary to proceed with actions that should only involve trusted ADVs. @grd06 ensures that ADV j has a valid entry in the adv variable, i.e., it has a known trust status (TRUST, NOT_TRUST). Without this, referencing adv(j) would be undefined, possibly causing a well-definedness proof of obligation in Event-B. @grd07 confirms that the pair (j ↦ t) meaning ADV j is responsible for task t exists within the range of the trustor−to−trustee−task mapping. It shows that some agent has explicitly trusted ADV j for task t, and this trust is recorded in the relation trustor_trustee_task. This ensures that j is not just generally trusted, but trusted specifically for this task. In this refinement, the trust event enables

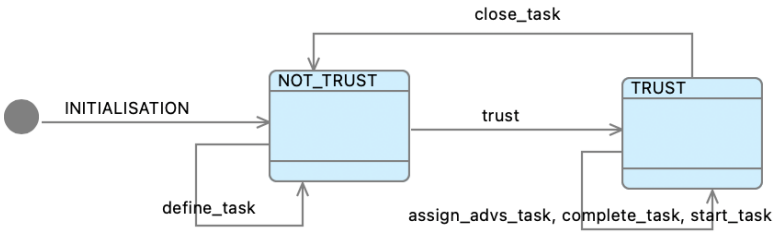

Fig. 2 State machine diagram illustrating the second refinement, which classifies ADVs as trusted or not based on their capabilities

the agent to establish reliance on an ADV, based on its verified ability to complete the task successfully guard from @grd01 to @grd06.

At this stage, trust is interpreted in terms of capability: an ADV is considered trustworthy if it has the means to complete the task, regardless of whether the agent is aware of this capability or whether the ADV has committed to performing the task.

4.2 3rd Refinement: Epistemic Trust

The third refinement introduces the concept of epistemic trust, which ensures that the agent assigns a task to an ADV only if it has sufficient knowledge of the ability of the ADV to perform that task. This refinement addresses the second failure scenario from the abstract model, where an agent may assign a task to an ADV whose capabilities are suitable in principle, but are unknown or unverifiable to the agent at the time of delegation. In the absence of verified knowledge, the assignment decision becomes speculative and may lead to failures if the agent's assumptions are incorrect.

$$@inv04 : knowledge \in agents \leftrightarrow advs$$
$$@inv05 : \forall i, t \cdot i \in agents \land t \in tasks \land t \notin dom(definedTask) \land t \in ran(trustees\_task) \Rightarrow (\exists j \cdot j \in dom(trustees\_task) \land trustees\_task(j) = t \land requiresMin[\{t\}] \subseteq capabilities[\{j\}] \land j \in knowledge[\{i\}])$$

To formally capture epistemic trust, we refine the model by introducing new invariants that represent the agent's knowledge of the ADVs' capabilities and by extending relevant events to reflect this trust dimension.

Invariant inv04 defines knowledge as a binary relation between agents and ADVs, capturing which ADVs are known to which agents. Invariant inv05 ensures that an agent delegates a task only if it knows an ADV whose capabilities satisfy the task's minimum requirements. Specifically, for any task not yet defined but already present in the trustees list, there must exist a trustee ADV whose capability set meets the task's requirements and is known to the delegating agent. Together, these invariants guarantee that task delegation is not only strategically feasible but also epistemically justified based on the agent's knowledge.

This refinement also affects the behavior of the trust event by introducing grd08, which restricts task assignment to only those ADVs that the agent knows are capable. In this way, trust is grounded in verified knowledge rather than assumption. Even if an ADV meets the strategic requirements, it cannot be selected unless the agent possesses

the epistemic justification to trust it. By enforcing epistemic trust, the model ensures that task assignments are knowledge-aware, thereby eliminating delegation based on uncertainty.

4.3 4th Refinement: Commitment trust

The fourth refinement introduces commitment trust, which ensures that an ADV is assigned a task only if it is intentionally committed to carrying it out. This addresses the third failure scenario identified in the abstract model, where an ADV may be capable and known to the agent but is either burdened with existing tasks, experiencing operational failure, or lacking sufficient time to complete the assigned task.

$$
\begin{aligned}
&@inv06 : commitments \in trustor\_trustee\_task \to BOOL \\
&@inv07 : \forall i, t \cdot i \in agents \land t \in tasks \land taskStart(t) = TRUE \Rightarrow (\exists j \cdot j \in advs \land \\
&(j \mapsto t) \in assignedTask \land \\
&adv(j) = TRUST)
\end{aligned}
$$

To formally capture commitment trust, the model is extended with a new state component that tracks the commitment status of each ADV. This includes whether an ADV is explicitly committed to executing a given task, describe in Figure 3. Invariant inv06 defines commitments as a total function from the set trustor_trustee_task to BOOL. This means that for every trust relationship between an agent (the trustor) and an ADV (the trustee) with respect to a task, there is an associated Boolean value indicating whether the ADV is committed TRUE or not FALSE to performing that task. Invariant inv07 ensures that any task marked as started (i.e., $taskStart(t) = TRUE$) must meet two conditions:

1. It is assigned to some ADV j, such that $(j \mapsto t) \in assignedTask$, and
2. ADV j must currently be in the TRUSTED state, i.e., $adv(j) = TRUST$.

This invariant guarantees that no task can be initiated unless it has been formally assigned to a trusted ADV. It enforces the core principle of commitment trust: task execution may only begin if the vehicle is explicitly trusted and has committed to carrying out the task. This reduces the risk of premature or unreliable task starts.

At this level, the trust event reflects a comprehensive trust relation: the agent may only trust an ADV if it is (1) capable of performing the task (strategic trust), (2) known to be capable (epistemic trust), and (3) available and committed to carry it out (commitment trust). This layered model of trust ensures that task assignments are not only feasible and informed but also actionable and reliable at the moment of delegation.

4.4 5th Refinement: ADV's States

The fifth refinement ensures the state consistency of trust notions as the ADV's state is updated during operation. While previous refinements introduced capability, knowledge, and commitment trust dimensions, this refinement verifies that those invariants continue to hold across dynamic ADV states such as available, ready, delivering, and returning.

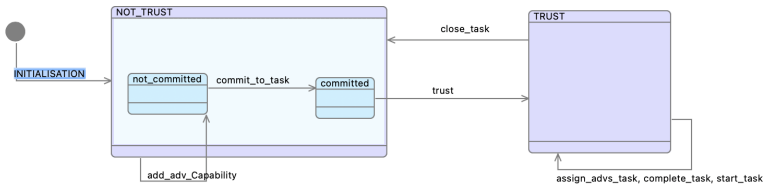

Fig. 3 State machine diagram illustrating the fourth refinement, which enforces Commitment Trust in ADV task delegation

$$
\begin{aligned}
&@inv08 : \forall i, j, t \cdot i \in agents \land j \in advs \land t \in tasks \land \\
&j \in dom\,(trustees\_task) \land trustees\_task(j) = t \land adv(j) = TRUST \\
&\Rightarrow j \in knowledge[\{i\}] \land i \mapsto (j \mapsto t) \in dom(commitments) \land \\
&commitments(i \mapsto (j \mapsto t)) = TRUE \land (i \mapsto (j \mapsto t)) \in trustor\_trustee\_task
\end{aligned}
$$

In this refinement, the model is extended to incorporate ADV state transitions that reflect the operational scenario depicted in Figure 4. These transitions introduce potential changes in availability and engagement, which may affect the trust assumptions made at the time of task assignment. Therefore, it becomes essential to prove that previously established invariants remain preserved as the ADV's state changes over time.

To formally guarantee this, we refine relevant events such as start_task, complete_task, trust, and close_task, to ensure they do not violate trust-related invariants. This involves checking that:

Task assignments remain valid as ADVs move between states;
No tasks are lost or reassigned during transitions;
The ADV's trust status remains aligned with its actual state.

By maintaining trust invariant preservation throughout the ADV's lifecycle, this refinement reinforces the model's robustness in practical deployment scenarios, where trust must be preserved.

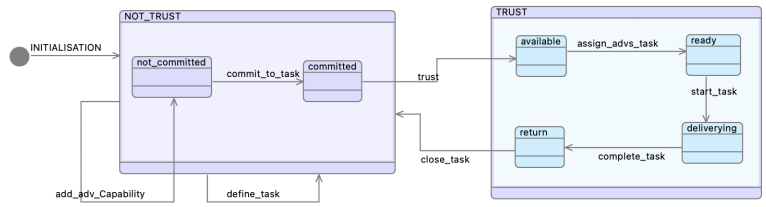

Fig. 4 State machine diagram illustrating the fifth refinement, where trust-related invariants are preserved across dynamic ADV states such as delivering and returning

5 Verification and Proof of Trust Properties

The trusted ADV Event-B model is verified using theorem proving techniques embedded in the Event-B toolset, called Rodin [18]. These proofs ensures the correctness of the model and consistency between different refinements, especially in terms of enforcing and preserving trust properties.

The Rodin tool automatically generates a set of proffg obligation(POs) to verfiy:

- invariant preservation (e/v/INV) ensures that every invariant v holds after execution of event e.
- The guard strengthening (e/g/GRD) ensures that the guard of refined (concrete) event is stronger than those of the abstract event, preserving behavioral consistency.

As an example of POs, the assign_task/inv08/INV ensures that tasks are only assigned to ADVs that are explicitly marketed as trusted. Here, the invariant inv08 requires that a trusted ADV must be in a valid operational state (e.g, `available`), and this is enforced in the assign_task event through guard grd10. Thus, trust is not just assumed but it is formally required and verified.

- **Quantified Invariants**: Such as ensuring that for all tasks, a corresponding ADV in a trusted state exists trust/inv08/INV. These often cannot be resolved automatically due to variable scoping or dependency on other events.
- **Guard-Strengthening**: For example, in the fifth refinements, where the ADV's state dynamically changes, guard strengthening POs were manually proven to ensure that trust-preserving behavior holds across states like `delivering`, `returning`, etc.

Table 1 Proving effort summary

Refinement name	Total	Auto	Manual
trust_ADV	131	117	13
M0_tasks	20	20	0
M1_agents	17	17	0
M2_ADV_trust	29	27	2
M3_ADV_knwl	10	7	3
M4_ADV_com	31	24	6
M5_ADV_states	24	22	2

As summarized in Table 1, a total of 131 POs were generated throughout the refinement process. Approximately 90% were discharged automatically by Rodin. The remaining 13 POs were proven manually, underscoring the rigor required to ensure that the model accurately captures trust-aware behavior under refinement.

These proofs collectively demonstrate that the model preserves trust invariants through refinements. The correctness of these invariants ensures the safety, predictability, and reliability of autonomous delivery vehicle delegation in untrusted environments.

6 Related Work and Discussion

Trust modeling in autonomous systems draws from formal methods, multi-agent systems, and human-computer interaction research. Traditional approaches have relied on statistical and reputation-based models [1, 19], with recent work shifting toward dynamic trust frameworks using simulation models [20] and machine learning [21]. However, these approaches primarily focus on empirical observation rather than formal verification guarantees. Formal methods for autonomous systems have gained significant attention [22], with formal verification recognized as essential for regulatory acceptance [23]. Dennis et al.[24] demonstrate formal verification of ethical decision-making, extending formal methods beyond traditional safety properties. Trust research in multi-agent systems provides foundational concepts through cognitive, game-theoretic, and socio-cognitive paradigms[25]. Recent advances include Trust Computation Tree Logic (TCTL) for formal trust reasoning [26] and combined trust-commitment verification frameworks [27]. However, these approaches address trust between software agents rather than trust in autonomous capabilities of physical systems. Event-B has been successfully applied to safety-critical domains [28], context-aware systems [29], and enhanced through graphical tools [30] and hybrid system extensions [31]. Despite this progress, several gaps remain: most trust models rely on probabilistic rather than formal guarantees, traditional approaches model trust as static rather than dynamic, existing models focus on historical performance rather than current capabilities, and integration of trust reasoning with system verification remains limited.

6.1 Contribution and Novelty

Our work addresses these gaps by introducing a formal framework for "actual trust" that evaluates current system state and capabilities rather than historical performance. Unlike statistical models, our Event-B formalization provides mathematical guarantees through theorem proving. The key contributions include: (1) a novel three-dimensional trust model integrating strategic, epistemic, and commitment trust; (2) a systematic refinement-based approach ensuring correctness-by-construction; and (3) formal verification of trust properties with 93 proof obligations achieving 90% automatic verification.

6.2 Evaluation and Limitations

The verification results demonstrate the feasibility of our approach, with the Rodin theorem prover successfully handling the majority of proof obligations automatically. The 11 manual proofs primarily involved complex invariant preservation across state transitions, indicating areas where automated reasoning reaches its limits. Compared to similar Event-B developments, our proof effort is reasonable for the complexity of trust reasoning integrated with autonomous system behavior. However, several limitations exist. First, our evaluation relies solely on formal verification without empirical validation in real deployment scenarios. The model assumes perfect communication

and does not address potential security threats or malicious agents. The trust dimensions are currently treated as boolean properties rather than continuous measures, which may limit applicability in scenarios requiring nuanced trust assessment.

The scalability of our approach presents both opportunities and challenges. The refinement-based methodology scales well to additional trust dimensions or more complex ADV behaviors through systematic model extension. However, the computational complexity of proof verification may increase significantly with larger fleets or more sophisticated trust reasoning. Future work should investigate distributed verification approaches and automated proof strategy generation. For practical deployment, our framework provides a foundation for trustworthy task delegation but requires integration with runtime monitoring and adaptation mechanisms. The static nature of our current model could be extended with dynamic trust update mechanisms that respond to changing operational conditions while preserving verified trust properties.

7 Conclusion and Future Direction

The study evaluates the trustworthiness of autonomous driving vehicles (ADVs) by assessing their perceptions according to actual trust notions. It uses Event-B to represent ADVs's capabilities, awareness of their ability to achieve the task, and knowledge of their current state at different refinement levels. The evaluation reveals that incorporating more information into the refinement, such as the ADVs capabilities and their status within the multi-agency context, enhances the possibilities of achieving the task and developing trust.

While modelling used to evaluate actual trust, it can also be embedded in task allocation systems for task assignment to trustworthy delivery vehicles, integrated into smart user assistants for task evaluation, and embedded in organisational settings for task delegation when a vehicle struggles to deliver due to unexpected issues.

Modelling of actual trust formally in ADVs can be expanded in multiple directions. One direction is to investigate cases in which achieving a task requires collaboration among multiple ADVs, hence the need for verification of actual trust on the coalitional level. Further research can also investigate modelling actual trust in ADV organisations with a hierarchical structure and the possibility of task delegation among ADVs. In such settings, we envisage that trust may propagate through the chain of task delegation.

Several research directions emerge from this work. Extending the framework to multi-ADV collaboration scenarios would require modeling coalitional trust and distributed decision-making. Integration with machine learning components for dynamic capability assessment while maintaining formal guarantees presents an interesting challenge. Additionally, developing domain-specific trust models for other autonomous systems (aerial vehicles, maritime systems) could demonstrate the broader applicability of our approach. The relationship between formal trust verification and regulatory compliance deserves investigation, particularly how our mathematical guarantees can support certification processes for autonomous delivery systems. Finally, empirical studies comparing our approach with existing trust models in real-world scenarios would provide valuable validation of the practical benefits of formal trust modeling.

References

[1] Kohn, S.C., De Visser, E.J., Wiese, E., Lee, Y.-C., Shaw, T.H.: Measurement of trust in automation: A narrative review and reference guide. Frontiers in psychology **12**, 604977 (2021)

[2] Afroogh, S., Akbari, A., Malone, E., Kargar, M., Alambeigi, H.: Trust in AI: progress, challenges, and future directions. Humanities and Social Sciences Communications, 11 (1) (2024)

[3] Barbosa, R., Santos, R., Novais, P.: Trust-based negotiation in multiagent systems: A systematic review. In: International Conference on Practical Applications of Agents and Multi-Agent Systems, pp. 133–144 (2023). Springer

[4] Ramchurn, S.D., Huynh, T.D., Jennings, N.R.: Trust in multi-agent systems. Knowl. Eng. Rev. **19**(1), 1–25 (2004)

[5] Bentahar, J., Drawel, N., Sadiki, A.: Quantitative group trust: A two-stage verification approach. In: Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems, pp. 100–108 (2022)

[6] Drawel, N., Laarej, A., Bentahar, J., El Menshawy, M.: Transformation-based model checking temporal trust in multi-agent systems. Journal of Systems and Software **192**, 111383 (2022) https://doi.org/10.1016/j.jss.2022.111383

[7] Salehi Fathabadi, A., Yazdanpanah, V.: Trust modelling and verification using event-b. (2023)

[8] Abrial, J.-R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press, ??? (2010)

[9] Halpern, J.Y.: Actual Causality. MiT Press, ??? (2016)

[10] Snook, C.F., Butler, M.J.: UML-B: A Plug-in for the Event-B Tool Set. In: Börger, E., Butler, M.J., Bowen, J.P., Boca, P. (eds.) Abstract State Machines, B and Z, First International Conference, ABZ 2008, London, UK, September 16-18, 2008. Proceedings. Lecture Notes in Computer Science, vol. 5238, p. 344. Springer, ??? (2008)

[11] Dghaym, D., Poppleton, M., Snook, C.: Diagram-led formal modelling using iuml-b for hybrid ertms level 3. In: Abstract State Machines, Alloy, B, TLA, VDM, and Z: 6th International Conference, ABZ 2018, Southampton, UK, June 5–8, 2018, Proceedings 6, pp. 338–352 (2018). Springer

[12] Dghaym, D., Hoang, T.S., Turnock, S.R., Butler, M., Downes, J., Pritchard, B.: An stpa-based formal composition framework for trustworthy autonomous maritime systems. Safety science **136**, 105139 (2021)

[13] Akintunde, M., Yazdanpanah, V., Salehi Fathabadi, A., Cirstea, C., Dastani, M., Moreau, L.: Actual trust in multiagent systems. In: Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024). IFAAMAS '24, pp. 2114–2116. International Foundation for Autonomous Agents and Multiagent Systems, Auckland, New Zealand (2024). https://doi.org/10.5555/3635637.3663078 . Extended Abstract

[14] Hoang, T.S.: An introduction to the Event-B modelling method. Industrial Deployment of System Engineering Methods, 211–236 (2013)

[15] Said, M.Y., Butler, M., Snook, C.: A method of refinement in uml-b. Software & Systems Modeling **14**(4), 1557–1580 (2015)

[16] Snook, C.: iuml-b statemachines new features and usage examples. In: Proceedings of the 5th Rodin User and Developer Workshop. University of Southampton, ??? (2014). https://eprints.soton.ac.uk/365301/

[17] Snook, C., Butler, M.: Uml-b: Formal modeling and design aided by uml. ACM Transactions on Software Engineering and Methodology (TOSEM) **15**(1), 92–122 (2006)

[18] Abrial, J.-R., Butler, M., Hallerstede, S., Hoang, T.S., Mehta, F., Voisin, L.: Rodin: An open toolset for modelling and reasoning in Event-B. Software Tools for Technology Transfer **12**(6), 447–466 (2010) https://doi.org/10.1007/s10009-010-0145-y

[19] Hoff, K.A., Bashir, M.: Trust in automation: integrating empirical evidence on factors that influence trust. Human Factors **57**(3), 407–434 (2015)

[20] Poornikoo, M., Gyldensten, W., Vesin, B., Øvergård, K.I.: Trust in automation (tia): Simulation model, and empirical findings in supervisory control of maritime autonomous surface ships (mass). International Journal of Human-Computer Interaction, 1–18 (2024)

[21] Zhang, X., Wang, Y., Li, J.: Human-autonomous teaming framework based on trust modelling. IEEE Transactions on Human-Machine Systems **52**(6), 1273–1284 (2022)

[22] Luckcuck, M., Farrell, M., Dennis, L.A., Dixon, C., Fisher, M.: Formal specification and verification of autonomous robotic systems: A survey. ACM Computing Surveys **52**(5), 1–41 (2019)

[23] Farrell, M., Luckcuck, M., Fisher, M.: Robotics and integrated formal methods: Necessity meets opportunity. In: Integrated Formal Methods. Lecture Notes in Computer Science, vol. 11023, pp. 161–171. Springer, ??? (2018)

[24] Dennis, L., Fisher, M., Slavkovik, M., Webster, M.: Formal verification of ethical

choices in autonomous systems. Robotics and Autonomous Systems **77**, 1–14 (2016)

[25] Pinyol, I., Sabater, J.: Computational trust and reputation models for open multi-agent systems: A review. Artificial Intelligence Review **40**(1), 1–25 (2013)

[26] Drawel, N., Laarej, A., Bentahar, J., El Menshawy, M.: Specification and automatic verification of trust-based multi-agent systems. Future Generation Computer Systems **107**, 1047–1060 (2020)

[27] Bentahar, J., Drawel, N., Sadiki, A.: Model checking combined trust and commitments in multi-agent systems. In: Proceedings of the 22nd International Conference on Autonomous Agents and Multiagent Systems, pp. 100–108 (2023). ACM

[28] Singh, N.K., Aït-Ameur, Y., Pantel, M., Dieumegard, A., Jenn, E.: Formal domain-driven system development in event-b: Application to interactive critical systems. Journal of King Saud University-Computer and Information Sciences **35**(9), 101722 (2023)

[29] Le, H.A., Truong, N.T.: Formal modeling and verification of context-aware systems using event-b. In: Proceedings of the 9th International Conference on Evaluation of Novel Software Approaches to Software Engineering, pp. 133–141 (2014). SCITEPRESS

[30] Karmakar, R., Datta, S.: A graphical tool for formal verification using event-b modeling. Multimedia Tools and Applications, 1–24 (2023)

[31] Banach, R.: Autonomous system safety properties with multi-machine hybrid event-b. In: Proceedings of the Sixth International Workshop on Formal Methods for Autonomous Systems, pp. 45–62 (2024). EPTCS