The Architecture of AgenTRIZ, a Multi-Agent LLM Framework for TRIZ Automation

Samuel Jonas Wenk[1,3][0009−0009−8417−6831], Jens Träger[2][0000−0003−3522−9000], Jürgen Moosburger[3][0009−0004−3018−4282], and Timo Baumann[1][0000−0003−2203−1783]

[1] OTH Regensburg, Germany
[2] Truinorva GmbH, Hetlingen, Germany
[3] ams-OSRAM International GmbH, Regensburg, Germany

Abstract. Large language models can support TRIZ-based inventive problem solving, as shown in recent work, yet existing systems operate as stateless single-pass pipelines. They cannot enforce the analytical sequence, are prone to hallucinating Contradiction Matrix entries, and have no access to project-specific engineering data. We present the architecture of AgenTRIZ, a multi-agent LLM system that covers the full TRIZ Abstraction Prism, from structured problem intake through functional analysis, root cause analysis, and contradiction formulation to matrix-guided solution synthesis, with human validation gating each stage transition. Three architectural mechanisms address the limitations of prior work. First, a Directed Cyclic Graph with hard prerequisite gates enforces the TRIZ dependency chain through topology rather than prompt-level instructions. Second, Contradiction Matrix lookups are deterministic tool calls against a verified dataset, eliminating the hallucination problem documented in prior work. Third, hybrid retrieval-augmented generation grounds solution synthesis in the engineer's own project documentation. A confirmation-gated commit protocol ensures that every analytical artefact is human-validated before it enters the persistent record. Evaluation comprises a comparison with a certified Level 3 TRIZ practitioner (zero methodological errors), a baseline comparison showing a statistically significant advantage for the multi-agent system (Wilcoxon $W = 105$, $p = 0.0005$, $r = 0.88$), two industrial semiconductor case studies, and component-level benchmarks. A resource analysis across 332 LLM calls confirms that reasoning capacity concentrates at the analytical stages where the architecture yields its largest quality advantage.

Keywords: TRIZ · Multi-Agent Systems · Large Language Models · Agentic AI · Computer-Aided Innovation

1 Introduction

The Theory of Inventive Problem Solving (TRIZ), developed by Altshuller [1] from the analysis of over 40,000 patents, provides a structured methodology for resolving engineering contradictions through the Abstraction Prism: a specific problem (P_{spec}) is abstracted to a generic problem (P_{gen}), mapped to generic

solutions (S_{gen}) via the Contradiction Matrix, and then specialised back to the project context (S_{spec}). It's power lies in this systematic abstraction, but it demands significant expertise to execute. Level 3 certification, the minimum for independent industrial application, requires completion of all three MATRIZ training levels plus a supervised project [18]. As a result, an expertise gap between the methodology's potential and its practical reach can be observed persistently.

Large language models have opened a new space for closing this gap. Systems such as AutoTRIZ [10] and TRIZ-GPT [3] demonstrate that LLMs can handle the upward translation ($P_{\text{spec}} \rightarrow P_{\text{gen}}$) with substantially better coverage than previous, keyword-based Semantic TRIZ approaches (see next section). However, current LLM-based systems share three structural limitations. First, they operate as stateless single-pass pipelines that pass a string from one stage to the next, with no mechanism to enforce the analytical dependency chain or to loop back when a later finding invalidates an earlier artefact. Second, they query the Contradiction Matrix from the model's parametric memory, producing plausible but frequently incorrect entries; Mysior and Cavallucci [20] report hallucination rates of 20–40 % for parameter identification alone. Third, they generate solutions without access to the engineer's project documentation, leaving the downward translation ($S_{\text{gen}} \rightarrow S_{\text{spec}}$) ungrounded.

We present AgenTRIZ, a multi-agent LLM architecture that covers the full Abstraction Prism, with human validation gating each stage transition, by addressing all three limitations through structural constraints rather than prompt engineering. The system assigns five specialist agents to the TRIZ analytical stages, coordinated by a centralised orchestrator within a Directed Cyclic Graph that enforces prerequisites through hard gates, performs matrix lookups deterministically via tool calls against a verified dataset, and grounds solution synthesis in the engineer's documentation through hybrid retrieval-augmented generation.

This paper makes the following contributions:

1. A multi-agent architecture with prerequisite enforcement through graph topology, where the TRIZ dependency chain is guaranteed by structural constraints in a Directed Cyclic Graph rather than by prompt-level instructions, and deterministic matrix lookup via tool calls which eliminates the hallucination problem documented in prior work.
2. A hybrid grounding approach that combines on-demand retrieval-augmented generation (semantic vector + BM25 keyword search) with a persistent artefact store and confirmation-gated commit protocol (Draft-Save), ensuring that solution synthesis is anchored in project-specific engineering data and that every committed artefact is human-validated.
3. A multi-level evaluation comprising an expert benchmark, a statistically significant baseline comparison, two industrial case studies, and component-level benchmarks with cost analysis.

The following Section 2 surveys computational TRIZ approaches and positions AgenTRIZ against recent TRAI work. Section 3 presents the system architecture. Section 4 reports the evaluation results. Section 5 discusses limitations and ethical considerations, and Section 6 concludes with directions for future work.

2 Related Work

Computational support for TRIZ has evolved through four generations, each closing part of the expertise gap while leaving residual limitations (Table 1).

Classical Computer-Aided Innovation (CAI) systems digitised the reference materials: the Contradiction Matrix, the 40 Inventive Principles, and functional modelling templates became searchable electronic databases [19]. These systems were passive. The engineer still had to formulate the contradiction and identify parameter numbers manually; the upward translation boundary remained entirely unaddressed.

Semantic TRIZ (S-TRIZ) integrated natural language processing into the CAI framework, enabling automatic extraction of Subject-Action-Object functional relationships from engineering text and patent corpora [8]. Part-of-speech tagging, dependency parsing, and lemmatisation allowed S-TRIZ systems to mine large document collections and map extracted SAO triples to TRIZ contradiction patterns without manual pre-processing. Two structural limitations persisted: the rigid Noun-Verb-Noun SAO structure discards operational context, system boundaries, and supersystem interactions critical for correct contradiction formulation; and the pattern-matching algorithms of the time could retrieve relevant historical examples but could not generate new solution concepts for problems outside the training corpus. The downward translation boundary remained unaddressed. Graph-based extraction approaches [26] partially address the first limitation by replacing the flat SAO triple with a structured problem graph that better preserves system-boundary relationships, though without generative solution synthesis or multi-agent coordination.

The arrival of LLMs with strong in-context reasoning made generative TRIZ automation feasible. TRIZ-GPT [3] and AutoTRIZ [10] both implement a four-stage linear pipeline: extract problem parameters, map them to TRIZ engineering parameters, query the Contradiction Matrix, and generate solution elaborations. Both showed that LLMs handle the parameter-mapping step with better coverage than keyword-based S-TRIZ, partially closing the upward translation gap. Three limitations remain in this architecture: (1) TRIZ analysis is iterative and a linear pipeline that passes only a string between stages cannot support revision loops; (2) querying the Contradiction Matrix from parametric memory produces plausible but frequently incorrect entries [20]; and (3) without external memory, system constraints established early in the reasoning chain are corrupted as the prompt grows [14]. AutoTRIZ includes a deterministic matrix-lookup step that addresses limitation (2), but its pipeline remains linear and stateless. Essaber and Cavallucci [7] corroborate this finding in a supply chain context, showing that AI-based contradiction solving suffers from the same parametric-memory unreliability across application domains beyond engineering product development.

Multi-agent architectures distribute the workflow across specialist agents coordinated by an explicit state machine. TRIZAgents [23] implements a supervised team of seven domain-specialist agents plus a Project Manager orchestrator on LangGraph, covering the full Abstraction Prism in six sequential steps. A Documentation Specialist records each step's output, providing partial state

Table 1. Comparison of LLM-based TRIZ automation approaches.

	AutoTRIZ [10]	TRIZ-GPT [3]	AICON [2]	TRIZAgents [23]	**AgenTRIZ**
Multi-agent	–	–	–	✓	✓
Full TRIZ workflow	✓	✓	–	✓	✓
Deterministic lookup	✓	–	–	–	✓
Document grounding	–	–	–	–	✓
Human-in-the-loop	–	–	✓	–	✓

continuity between stages. The TRIZ Specialist agent has access to a classical 39-parameter matrix lookup tool, but tool usage is not architecturally enforced; the authors note that the agent frequently relied on internal knowledge instead. The system's main acknowledged limitation is the absence of a feedback loop: the Project Manager never routes execution back to an earlier step to refine an artefact, so revision cycles are not supported. AICON [2] takes a different approach, focusing on the contradiction resolution step rather than the full prism. It uses retrieval-augmented generation to access diverse knowledge domains and identify novel inventive principles beyond the classical matrix entries. AICON includes a human-in-the-loop confirmation mechanism but operates as a single-model system without multi-agent coordination or project-specific document grounding.

Table 1 compares five capabilities relevant to LLM-based TRIZ automation. Four are present in at least one prior system; document grounding is new. AgenTRIZ is the first to combine all five. Beyond the tabulated features, prerequisite enforcement through hard gates in a Directed Cyclic Graph ensures that the TRIZ dependency chain is maintained structurally rather than through prompt-level instructions, and a persistent artefact store retains all committed analytical results across sessions.

3 System Architecture

The system topology follows directly from how TRIZ works: ISQ data must be on record before functional analysis starts; root causes have to be committed before the contradiction step is unlocked; the matrix lookup cannot be a generative guess. These are ordering constraints, not preferences, and they need structural enforcement. A linear prompt chain cannot provide that. Every design decision in this section traces back to one of those constraints.

3.1 Formal Agent Topology

The system is implemented as a Directed Cyclic Graph (DCG) compiled with LangGraph [6] (Figure 1). Five specialist agent nodes run the TRIZ workflow, one per analytical stage, coordinated by a centralised orchestrator. The TRIZ methodology imposes a partial order on the analytical stages:

$$v_{gather} \prec v_{func} \prec v_{rca} \prec v_{contra} \prec v_{solver} \tag{1}$$

Three conditional edge types (`should_continue`, `tool_route`, and the orchestrator's `next_step`) govern execution at runtime. Forward edges advance

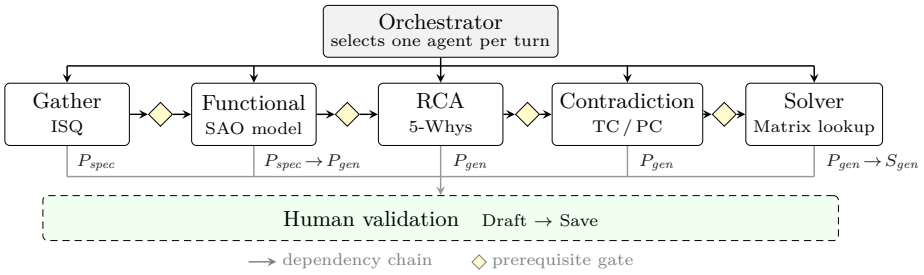

Fig. 1. Directed Cyclic Graph topology. The orchestrator routes each user turn to one specialist node. Diamond gates enforce the TRIZ dependency chain: a downstream node is only reachable when all upstream artefacts have been committed. Backward routing to an earlier node is possible through the orchestrator when a later finding invalidates an earlier artefact. Every agent output passes through human validation (Draft-Save) before entering the persistent record. Italic labels show positions in Abstraction Prism.

the analysis; backward edges enable revision loops when a later finding invalidates an earlier artefact, for example an RCA result that requires revising the functional model. Prerequisite enforcement is a hard gate in the orchestrator: if the computed routing decision would invoke `solver_node` but no committed contradiction exists, the decision is overridden to `contradiction_node`. The gates fire regardless of what the user asked for, blocking any attempt to generate solutions from an incomplete analytical base.

Routing decisions are produced via Pydantic-validated structured output, where the `next_step` field is a Python `Literal` covering only valid node names. The model cannot name a node that does not exist.

3.2 Structured Problem Intake

One entry point to the Prism is the Innovation Situation Questionnaire (ISQ). Several formulations exist; the classical form is due to one of the standard introductory textbooks on TRIZ [24]. AgenTRIZ does not copy these but adapts them instead. The `gather_node` populates a restructured schema of 22 fields designed to give a 360-degree view of the system covering its components and supersystem, useful and harmful interactions, inputs and outputs, conflicting requirements, available resources and constraints, and success metrics. The schema is intentionally a hybrid that bundles problem definition with preparation for functional analysis, root cause analysis, and contradiction formulation, so that a single machine-parseable intake can extract seeds for all four downstream artefacts (function model, RCA, and technical and physical contradictions). Typing every field as a Pydantic schema also keeps the captured constraints in the persistent store rather than in conversational prose, making them immune to the summarization loss reported in Section 5.1. This makes the intake closer to a structured project-intake questionnaire than to a full classical ISQ.

3.3 Agent Design and SOPs

Each specialist agent receives a Standardised Operating Procedure (SOP) as its system prompt, following Hong et al.'s [9] finding that encoding human SOPs directly into prompt sequences reduces logic inconsistencies when a model's generative tendencies conflict with methodological requirements. An SOP defines (a) the agent's TRIZ role and the concepts it must reason about, (b) contrastive correct/incorrect examples that anchor the output format, and (c) the exact Pydantic output schema the agent must produce.

The RCA agent's SOP, for instance, specifies an algorithmic loop: pick one harmful function from the committed `FunctionalModel`, trace it through iterative 5-Whys, and terminate only when the chain reaches a fundamental physical constraint or a management boundary. Stopping at symptoms is explicitly prohibited. The current implementation uses linear 5-Whys rather than the full Cause-Effect Chain Analysis (CECA) with AND/OR operators; this is sufficient for the linear causal chains in our evaluation but represents a scope limitation discussed in Section 5. The contradiction agent's SOP is equally rigid: engineering contradictions must follow IF-THEN-BUT form; physical contradictions must use [State A] AND [State B] notation. No other format is accepted. LangGraph manages tool access, therefore prompt-level instructions cannot interfere.

3.4 Deterministic Matrix Lookup

The Contradiction Matrix is a precise data structure: for each pair of engineering parameters, it returns a ranked list of Inventive Principles derived from patent analysis. AgenTRIZ uses the Matrix 2003 with its extended set of 48 parameters [15,16], rather than Altshuller's original. When an LLM queries this matrix from parametric memory, it produces plausible but frequently incorrect entries; Mysior and Cavallucci [20] report hallucination rates of 20–40 % for parameter identification alone. AgenTRIZ removes the matrix lookup from the generative path: `solver_node` invokes a `solve_technical_contradiction` tool that executes a programmatic table lookup in the Matrix 2003 dataset and returns the principle list for the given parameter pair; the LLM never generates these values.

The lookup follows a two-step process. First, a `get_triz_parameters` tool returns the full 48-parameter ID-to-name table, allowing the model to match project-specific engineering strings to canonical TRIZ parameter names by direct inspection rather than from memory. Second, `solve_technical_contradiction` takes the integer parameter IDs and performs a CSV row lookup, returning the ranked principle list with zero LLM inference involved. This separation means the generative model handles the conceptual mapping (which of the 48 abstract parameters best describes a project-specific engineering property?) while the verified data structure handles the factual retrieval (which principles does the matrix recommend for a given parameter pair?). The parameter mapping step remains a generative judgement call where different practitioners may select different abstract parameters for the same engineering property; the architecture ensures that once the mapping is made, the downstream lookup is exact.

Table 2. Token consumption by model tier across 332 evaluation calls.

Tier (model)	Token share	Primary nodes	Task type
Reasoning (o3)	47 %	RCA, Contradiction, Solver	Analytical core
Fast (gpt-5-mini)	44 %	Orchestrator, Gather, FuncAn	Extraction & routing
Default (gpt-5.2)	9 %	Chat interface	Conversation

3.5 Hybrid Retrieval-Augmented Generation

Recent work has shown that retrieval-augmented generation can extend AI-generated solution quality in inventive problem solving beyond parametric memory, including visual retrieval applied to problem formulation [17]. In AgenTRIZ retrieval is implemented as an on-demand agent tool (`search_project_documents`) Agents invoke RAG only when they identify a specific knowledge gap, conserving context window capacity and avoiding Lost-in-the-Middle degradation [14].

The tool executes hybrid search: dense semantic vector matching and sparse BM25 keyword matching run in parallel, with results merged via Reciprocal Rank Fusion (RRF). Both channels are necessary for TRIZ work. Semantic search captures conceptual matches across vocabularies: a query about "rotary drive mechanism" surfaces documents mentioning "motor and gearbox assembly." BM25 preserves exact technical nomenclature: when the contradiction agent needs to confirm that a specific resin has a yield stress above 150 Pa, or that the roller material is nitrile, dense embeddings for "high-friction polymer" would return conceptually adjacent but imprecise results. BM25 retrieves on exact tokens, ensuring precise technical values reach the contradiction and solver agents without dilution.

3.6 Resource-Aware Model Routing

A tiered routing strategy [12] assigns each node to one of three model classes according to task complexity: a fast model (gpt-5-mini) handles routing decisions, ISQ field extraction, SAO classification, and summarisation; a reasoning model (o3) handles root cause analysis, contradiction formulation, parameter identification, and solution synthesis; a general-purpose model (gpt-5.2) covers conversational turns and artefact drafting. The boundary maps onto the Abstraction Prism: upward translation extracts and structures existing input, the analytical core generates new knowledge through multi-step deduction.

Table 2 reports the resource profile across 332 LLM calls (3,606,215 tokens) recorded during the evaluation sessions. Despite covering fewer node types, the reasoning tier accounts for 47 % of total token consumption; the analytical core is where computational cost concentrates. The fast tier reaches 44 % through sheer invocation frequency: routing and extraction calls are individually cheap but numerous. Clarifications via the default tier accounts for the remaining 9 %.

3.7 Optimisation Perspective

The deterministic matrix lookup can be formalised as a constraint-satisfaction problem, connecting TRIZ automation to the broader landscape of AI-assisted optimisation for complex systems design. Given a set of contradictions $\mathcal{C} = \{(p_i^+, p_i^-)\}_{i=1}^n$, the matrix returns a principle set Π_i for each contradiction. When multiple contradictions coexist in a project (the typical industrial case), the engineer must select a subset of principles that addresses as many contradictions as possible:

$$\max_{S \subseteq \Pi,\ |S| \leq k} \sum_{c_i \in \mathcal{C}} \max_{\pi \in S} r(\pi, c_i) \tag{2}$$

where $r(\pi, c_i)$ combines the matrix rank (historical patent frequency) with LLM-assessed contextual relevance to the specific project. This is a weighted maximum coverage problem. AgenTRIZ currently solves it greedily, but the formalisation opens a path toward Pareto-optimal principle selection across competing contradictions, a direction we return to in Section 6.

4 Evaluation

Four evaluation strands assess the system across complementary dimensions: an expert comparison for methodological fidelity, a statistically tested baseline comparison for architectural value, two industrial case studies for practical utility, and component-level benchmarks for subsystem correctness.

4.1 Expert Benchmark

The second author, a certified Level 3 TRIZ practitioner, independently analysed a laboratory resin mixer and documented the full Abstraction Prism. The same problem description was submitted to AgenTRIZ with no access to the expert's document. Because TRIZ is a divergent methodology where two practitioners may formulate different but equally valid contradictions, a simple match/mismatch classification would be misleading. Therefore, and following [5,22], the comparison uses three categories: *Convergent Match* (both identified the same element), *Novel-Valid* (analytically correct but absent from the expert's analysis), and *Methodological Error* (a genuine violation such as a malformed contradiction or a fabricated matrix entry).

Table 3 shows the aggregate. Zero methodological errors were found across all six stages. Every contradiction follows IF-THEN-BUT structure with valid 48-parameter pairs; every matrix lookup is verifiable against the published Matrix 2003 because it was produced by a CSV tool call, not by the model's memory. Coverage gaps exist (one root cause thread, N/I/E fulfilment grading) but none are structural violations.

Table 3. Expert comparison: multi-agent system vs. certified Level 3 TRIZ practitioner on the resin mixer case study. Expert-only items are coverage gaps, not errors.

TRIZ Stage	Match	Novel-Valid	Expert-Only
Component Analysis	9/11	2	0
Functional Model[a]	6/8	3	0[c]
Root Cause Analysis	4/5	3	1
Eng. Contradictions	3/6	2	1
Phys. Contradictions	2/5	2	2
Solution Concepts[b]	4/9	2	3

Counted by [a]harmful interaction overlap; [b]thematic convergence across solution directions. [c]N/I/E grading absent/convention gap.

4.2 Baseline Comparison

To isolate the architectural and interactive contribution, we constructed a monolithic single-agent baseline using the same underlying LLM. The baseline received one prompt containing the complete TRIZ methodology, all 48 engineering parameters, the 40 Inventive Principles, and the full project context. It executed all five TRIZ stages in a single pass, without tools, memory, or document retrieval. The multi-agent system was operated by engineers across real multi-session interactions with validation gates at each stage boundary; the baseline produced a single output per project. This asymmetry reflects how the two architectures would be deployed in practice.

The outputs from both systems across three projects were scored by an LLM judge (o3, not involved in either system's generation pipeline) on five dimensions: completeness, TRIZ adherence, technical accuracy, context grounding, and solution quality (each 1–5). Both outputs were presented under anonymised labels with randomised assignment (cmp. [27]). The evaluation covers 15 deliverables (3 projects $\times$ 5 TRIZ stages).

Table 4 shows the stage-averaged scores. The largest gaps appear at the RCA and contradiction stages (4.27 vs. 3.13 and 4.40 vs. 3.13), precisely where the architecture provides its structural advantages: iterative causal chain traversal across multiple dialogue turns, deterministic matrix lookup, and prerequisite enforcement that prevents contradiction formulation from incomplete analytical data. ISQ scores are close (3.67 vs. 3.47) because both systems perform unconstrained text elicitation at that stage and neither benefits from the architectural constraints that differentiate them downstream.

A one-sided Wilcoxon signed-rank test on the 15 paired stage scores (H_1: multi-agent $>$ baseline) yields $W = 105$, $p = 0.0005$, with a large effect size ($r = 0.88$). One pair (Case A, ISQ) is excluded as tied ($d = 0$), leaving $n = 14$ non-zero differences; every non-tied pair favours the multi-agent system. The test establishes consistency of direction within this evaluation; generalisability to other projects or judges cannot be claimed from it alone.

The baseline explicitly acknowledged it could not perform the matrix lookup, falling back to principle selection from training data. Cross-checking confirmed that the multi-agent system's cited principles appear in the correct Matrix 2003 cells. The baseline's citations contained parameter vocabulary mismatches [20].

Table 4. LLM-judge scores averaged by TRIZ stage across all three projects.

Stage	Multi-Agent	Baseline
ISQ	3.67	3.47
Functional Model	4.07	3.27
Root Cause Analysis	4.27	3.13
Contradictions	4.40	3.13
Solutions	3.93	3.47

4.3 Industrial Case Studies

Two industrial engineering projects from the ams-OSRAM's semiconductor and optoelectronics divisions were processed through the full pipeline by domain engineers not involved in the system's development, to test performance of human-machine collaboration in real production problems (cmp. [11]).

Case A (semiconductor packaging process optimisation) addressed geometric distortion from thermomechanical mismatch. The system completed the full pipeline: 13 system and 10 supersystem components, 4 causal chains with 45 root causes, 6 technical and 4 physical contradictions, and 3 engineer-selected solution concepts. All matrix lookups are verifiable against the 2003 matrix.

Case B (automotive multi-pixel LED module) was substantially larger, comprising 84 system and 137 supersystem components, 20 functional interactions across optical, thermal, mechanical, and reliability domains, and 13 contradictions (10 TC + 3 PC).

Three engineers completed a nine-dimension Likert questionnaire (1–5). State retention scored highest (mean 5.0), consistent with the design guarantee of the persistence layer, usability scored 4.7. Trust and reliance scored lowest (mean 3.3), which is appropriate given the user should actively question outputs and enforce validation.

4.4 Component Benchmarks

Five subsystem benchmarks with deterministic ground truths complement the end-to-end evaluations. Orchestrator routing reaches 87.5 % accuracy across 200 test cases and six routing targets; four of the six routes score $\geq$96.9 %, with the solver route at 35.7 % as the outlier due to lexical overlap with contradiction-directed phrasing (Sec. 5.1). RAG retrieval achieves 100 % across 60 scenarios (20 documents), though the evaluation corpus is more clearly separated than a production repository. State-store retrieval (10 scenarios) and schema enforcement (50 scenarios) both hold at 100 %: no invalid artefact was persisted, and no agent bypassed the human validation gate. Short-term summarisation retains 70 % of conversational constraints across 10 scenarios; constraints captured through structured ISQ fields are immune to this loss. The per-node call distribution (Table 2, Sec. 3.6) confirms that the rca_node dominates analytical effort with 73 of 332 calls, consistent with its iterative 5-Whys traversal.

5 Discussion

5.1 Limitations

Two quantified weaknesses stand out. First, solver routing accuracy is limited at 35.7 %. Phrases like "solve this contradiciton" and "formulate a contradiction" share vocabulary and mislead the orchestrator. In production, the prerequisite gate prevents premature solver invocation so in practice human intervention avoids failure. However, three paths for improvement exist: (1) fine-tuning the routing model on a TRIZ-specific intent dataset with contrastive solver/contradiction examples, (2) introducing a two-step confirmation where the orchestrator first classifies intent and then verifies against the current analytical state before routing, and (3) augmenting the routing prompt with the committed artefact status so that the model can distinguish working on vs. solving for contradictions.

Second, short-term summarisation retains only 70 % of conversational constraints and technical values stated in prose (material grades, voltage ceilings) can be lost in context compression. Constraints captured through the structured ISQ schema are immune to this problem because they reside in the persistent store, not in the conversation history. The practical consequence is that critical constraints should be routed through structured fields rather than be summarized.

Beyond these component-level issues, the evaluation has structural limitations. The expert benchmark uses one practitioner on one problem; a different Level 3 practitioner would likely produce a different but equally valid analysis. The LLM-as-judge protocol assesses structural TRIZ correctness by reasoning, not against a verified matrix reference, and cannot evaluate engineering feasibility. The industrial case studies provide only a limited view.

The current RCA implementation uses iterative 5-Whys rather than full Cause-Effect Chain Analysis (CECA) with AND/OR logical operators. While 5-Whys is sufficient for the linear causal chains encountered in the evaluation projects, it cannot represent concurrent causation or branching failure modes [4]. Upgrading to CECA would bring the implementation closer to the methodology as practised at Level 3 and above, and constitutes a well-defined extension path.

5.2 Responsible AI and Intellectual Property

A final consideration concerns ethics and responsibility. As AgenTRIZ performs both Prism translations computationally, it must be clear where machine support ends and human responsibility begins. Current patent-office guidance defines inventorship as exclusively human [25]. AgenTRIZ's solver therefore is just a tool and the patentable contribution lies in the engineer's framing, parameter judgement, and selection. The Draft-Save pattern is a mechanism that makes this human conception demonstrable: every artefact is persisted only on explicit human confirmation, yielding an auditable, stage-keyed record that implements traceability and accountability [21]. A complementary concern in industrial contexts is freedom to operate (FTO). Even if a contradiction-resolving concept can expand the design space, the solutions found can still fall within third-party

claims. In most processes, such conflicts typically surface late and redirection is costly. However, a proactive and iterative FTO integration process [13] could in the future be integrated with AgenTRIZ.

6 Conclusion

This paper presented the architecture of AgenTRIZ, a LLM-based multi-agent-system that automates the TRIZ Abstraction Prism through structural constraints. Five specialist agents coordinated by a centralised orchestrator in a Directed Cyclic Graph enforce the TRIZ dependency chain via hard prerequisite gates, perform Contradiction Matrix lookups and ground solution synthesis in the engineer's project documentation through hybrid retrieval-augmented generation. The Draft-Save protocol ensures that every analytical artefact is committed only on explicit human confirmation and ensures the user remains the ultimate decision-maker.

Evaluation across four complementary strands found zero methodological errors in the expert comparison, a statistically significant advantage over a monolithic baseline ($W = 105$, $p = 0.0005$, $r = 0.88$) concentrated at the stages where the architecture provides structural enforcement (RCA: 4.27 vs. 3.13; Contradictions: 4.40 vs. 3.13), successful pipeline completion on two industrial semiconductor case studies, and component-level benchmarks confirming 87.5 % routing accuracy, 100 % retrieval fidelity, and 100 % schema enforcement.

Three directions for future work follow from the evaluation. First, solver routing accuracy (35.7 % in isolation) can be improved through contrastive fine-tuning or a two-step confirmation mechanism. Second, the linear 5-Whys implementation can be extended to full CESA. Third, the weighted maximum coverage formalisation introduced in Section 3.7 opens a path toward Pareto-optimal principle selection across competing contradictions, moving beyond the current greedy strategy. Beyond component improvements, the broader TRIZ toolkit (Substance-Field Analysis, ARIZ-85B, Trends of Engineering System Evolution) can be mapped onto the existing multi-agent topology as additional specialist nodes and FTO could be integrated.

The value of the multi-agent architecture lies not in making the LLM more capable, but in constraining it where unrestricted generation makes it unreliable. The system does not replace domain expertise; it structures the analytical process so that the engineer's judgement is applied at every step.

Acknowledgements and Disclosure of Interests. We thank ams-OSRAM for providing the industrial use cases. We declare not to have competing interests.

References

1. Altshuller, G.: The Innovation Algorithm: TRIZ, Systematic Innovation and Technical Creativity. Technical Innovation Center, Worcester, MA (1999)
2. Brad, S., Brad, E., Cîrlejan, A.: Enhancing TRIZ contradiction resolution with AI-driven contradiction navigator (AICON). In: TFC 2024. Springer (2025)

3. Chen, L., Song, Y., Ding, S., Sun, L., Childs, P., Zuo, H.: TRIZ-GPT: An LLM-augmented method for problem-solving. In: IDETC-CIE (2024)
4. Chrząszcz, J.: Systematic cause elimination in TRIZ cause-effect models with partially defined logical operators. In: TRAI 2025. Springer (2026)
5. Čok, V., Samsa, L., Brojan, M., Tavčar, J., Vukašinović, N.: Case study: Is there a space for TRIZ in the era of ChatGPT? In: Proc.s of the Design Society (2025)
6. Duan, Z., Wang, J.: Exploration of LLM multi-agent application implementation based on LangGraph+CrewAI. arXiv preprint arXiv:2411.18241 (2024)
7. Essaber, F.E., Cavallucci, D.: Artificial intelligence for contradiction solving in lean green supply chain performance context. In: TRIZ – The Theory of Inventive Problem Solving. Springer (2026)
8. Ghane, M., Ang, M.C., Cavallucci, D., et al.: Semantic TRIZ feasibility in technology development, innovation, and production. Heliyon **10**(1), e23775 (2024)
9. Hong, S., Zhuge, M., Chen, J., et al.: MetaGPT: Meta programming for a multi-agent collaborative framework. arXiv preprint arXiv:2308.00352 (2024)
10. Jiang, S., Li, W., Qian, Y., Zhang, Y., Luo, J.: AutoTRIZ: Automating engineering innovation with TRIZ and large language models. Adv. Eng. Inform. **65** (2025)
11. Kanarik, K.J., Osowiecki, W.T., Lu, Y., et al.: Human–machine collaboration for improving semiconductor process development. Nature **616**, 707–711 (2023)
12. Lee, C.H., Cheng, H., Ostendorf, M.: OrchestraLLM: Efficient orchestration of language models for dialogue state tracking. arXiv preprint arXiv:2311.09750 (2024)
13. Lehner, C., Träger, J., Cali, R., Wyder, T., Mayer, O.: Think. Solve. Create. Truinorva, Hetlingen, Germany (2026)
14. Liu, N.F., Lin, K., Chen, D., et al.: Lost in the middle: How language models use long contexts. TACL **12**, 157–173 (2024)
15. Mann, D.: Hands-On Systematic Innovation. CREAX Press, Ieper (2002)
16. Mann, D., Dewulf, S., Zlotin, B., Zusman, A.: Matrix 2003. CREAX, Ieper (2003)
17. Masúdah, Livotov, P., Hartmann, N., Nugroho, S., Kokoschko, B.R., Xu, W., et al.: Enhancing AI-generated solutions with visual retrieval-augmented generation (V-RAG). In: TRIZ – The Theory of Inventive Problem Solving. Springer (2026)
18. MATRIZ: Certification (2026), https://matriz.org/certification/
19. Moehrle, M.G.: How combinations of TRIZ tools are used in companies – results of a cluster analysis. R&D Management **35**(3), 285–296 (2005)
20. Mysior, M., Cavallucci, D.: Contradiction processing using large language models and generative artificial intelligence. In: TRIZ – The Theory of Inventive Problem Solving. IFIP AICT, vol. 774, pp. 171–183. Springer (2026)
21. OECD: AI principles overview. https://oecd.ai/en/principles, accessed 2026-05-21
22. Phadnis, N., Torkkeli, M.: Evaluating the effectiveness of generative AI in TRIZ: A comparative case study. In: TFC 2024. pp. 175–192. Springer (2025)
23. Szczepanik, K., Chudziak, J.A.: TRIZ agents: A multi-agent LLM approach for TRIZ-based innovation. In: Proceedings of the 17th International Conference on Agents and Artificial Intelligence. pp. 196–207 (2025)
24. Terninko, J., Zusman, A., Zlotin, B.: Systematic Innovation: An Introduction to TRIZ (Theory of Inventive Problem Solving). St. Lucie Press (1998)
25. USPTO: Revised inventorship guidance for AI-assisted inventions. Federal Register (2025)
26. Witkowicz, N., Cavallucci, D., Chibane, H.: Leveraging problem graph extraction and improvement perspectives for AI-assisted invention. In: TRIZ – The Theory of Inventive Problem Solving. Springer (2026)
27. Yehudai, A., et al.: Survey on evaluation of LLM-based agents. arXiv preprint arXiv:2503.16416 (2025)