

ARTICLE

TriGenDT: A hybrid TRIZ, design thinking, and
generative artificial intelligence framework
for systematic innovation in healthcare
management information systemsFatma Mohammed Dhaou*Management Information Systems Department, Faculty of Business Administration, University of
Tabuk, Tabuk, Saudi Arabia

Abstract

Innovation teams in regulated digital-service environments routinely encounter hard architectural contradictions, situations in which two non-negotiable requirements are in direct technical conflict. Despite significant scholarly interest in Theory of Inventive Problem Solving (TRIZ)-based contradiction resolution and the institutionalization of Design Thinking in software engineering and management information systems, no existing framework has formally coupled these two paradigms with the generative potential of large language models into a unified, human-centered innovation process. The current study presents TriGenDT, a five-stage Design Science Research artifact that integrates TRIZ contradiction resolution, Design Thinking, and generative artificial intelligence to enable systematic innovation in healthcare management information systems. The framework is illustrated through a constructed clinical decision-support scenario that highlights the tension between real-time alert latency and privacy/compliance constraints. The demonstration illustrates process coherence, stage-to-tool traceability, and design-level feasibility; it does not constitute field validation. Projected values, such as alert latency (~30 ms) and alert-path call-depth reduction (~50%, from six to three hops), are design-level estimates that require empirical confirmation. The study contributes a formally specified TRIZ-to-software-engineering parameter mapping, an integrated innovation methodology grounded in a reproducible demonstration case, a preliminary expert appraisal, representative prompt templates with model parameters, and an openly available artifact repository. Future research should pursue empirical validation through live industry case studies and comparative evaluation against TRIZ-only, Design Thinking-only, and large language model-native TRIZ baselines.

***Corresponding author:**
Fatma Mohammed Dhaou
(fdhaou@ut.edu.sa)

Citation: Dhaou, F. M. (2026).
TriGenDT: A hybrid TRIZ, design
thinking, and generative artificial
intelligence framework for
systematic innovation in healthcare
management information systems.
Int J Systematic Innovation, 10(4):
026120029.
[https://doi.org/10.6977/
IJoSI.202608_10\(4\).0002](https://doi.org/10.6977/IJoSI.202608_10(4).0002)

Received: March 20, 2026

Revised: June 6, 2026

Accepted: July 1, 2026

Published online: August 24, 2026

Copyright: © 2026 Author(s).
This is an Open-Access article
distributed under the terms of the
Creative Commons Attribution
License, permitting distribution,
and reproduction in any medium,
provided the original work is
properly cited.

Publisher's Note: AccScience
Publishing remains neutral with
regard to jurisdictional claims in
published maps and institutional
affiliations.

Keywords: TriGenDT; Theory of Inventive Problem Solving; Design Thinking; Generative artificial intelligence; Large language models; Management information systems; Design Science Research; Healthcare innovation

1. Introduction

Complex software system design and development are increasingly demanding rigorous problem formulation, human-centered requirement capture, and computationally

enhanced solution-generation methodologies. Management information systems (MIS) and software used in healthcare, one of the areas where regulatory burden, multi-stakeholder demand, and immediate patient safety concerns are typical, are particularly acute areas of this issue. Development teams often face requirement contradictions where two non-negotiable stakeholder demands directly oppose each other, and typical agile or waterfall methodologies have no structured way to resolve these conflicts (Aguilar-Calderón *et al.*, 2025).

The Theory of Inventive Problem Solving (TRIZ), proposed by Al'tshuller (1996) after systematic study of over 400,000 patents, provides the mechanism. Its contradiction matrix and 40 inventive principles form a knowledge-based system for determining and solving technical conflicts in a domain-independent way. The canonical expression of Design Thinking, as articulated by Brown (2008) and operationalized to software engineering contexts by Aguilar-Calderón *et al.* (2025), adds an organized, empathy-based front-end that grounds technical problem-solving in user needs. Generative artificial intelligence (AI), specifically instruction-tuned large language models (LLMs), has recently shown emergent functionality in natural-language contradiction detection (Sun *et al.*, 2026), cross-domain principle contextualization (Obojski, 2026), and functional system modeling (Mysior *et al.*, 2025) that could considerably reduce the expertise barrier to TRIZ adoption.

Each of the three paradigms belongs to independent scholarly consideration. Nevertheless, their formal consolidation into a single operationalizable and systematically specified methodology has not yet been studied. This lack is consequential: healthcare MIS teams face hard architectural contradictions, speed versus privacy, availability versus security, comprehensiveness versus compliance, and lack any systematic course of action between problem identification and validated solution. The current study fills this gap. The proposed study is based on four research questions (RQs):

- RQ1 (framework mapping): How can TRIZ contradiction-resolution concepts be systematically mapped to the stages of Design Thinking in a software engineering and MIS setting?
- RQ2 (AI augmentation): To what extent can generative AI (GenAI) and LLMs support contradiction identification and resolution within a TRIZ-integrated Design Thinking framework?
- RQ3 (comparative effectiveness): How does TriGenDT compare to TRIZ-only and Design Thinking-only approaches in terms of addressing MIS design contradictions?

- RQ4 (adoption factors): What are the critical success factors and adoption barriers for hybrid systematic innovation frameworks in MIS and software engineering teams?

This study has three contributions. First, it presents TriGenDT as a theoretically grounded conceptual framework that formally maps TRIZ contradiction-resolution phases to Design Thinking stages. Second, it provides an operational Design Science Research (DSR) artifact comprising a prototype tool, facilitation guide, and prompt templates. Third, it presents a constructed illustrative healthcare MIS case together with a preliminary expert appraisal providing initial plausibility evidence.

The remainder of this paper is organized as follows. Section 2 reviews the relevant literature. Section 3 presents the TriGenDT framework. Section 4 describes the research methodology and formalized evaluation framework. Section 5 presents the demonstration walkthrough. Section 6 discusses findings and introduces cross-domain applicability. Section 7 concludes with contributions and future research directions.

2. Literature review

This section reviews the scholarly foundations of TriGenDT across three streams: TRIZ and AI-augmented inventive problem solving (Section 2.1), Design Thinking in software engineering and MIS (Section 2.2), and human-centered AI (HCAI) as a normative foundation (Section 2.3). A formal articulation of the research gap and a comparative mapping table are given in Section 2.4.

2.1. Theory of Inventive Problem Solving and artificial intelligence-augmented inventive problem solving

Theory of Inventive Problem Solving was proposed by Al'tshuller (1996) through a systematic study of around 400,000 patents. The theory assumes innovation to be systematic rather than stochastic: technical contradictions between engineering parameters recur across industries, and the inventive principles that have historically resolved them can be systematically applied to new problems. The contradiction matrix, which maps pairs of conflicting engineering parameters to recommended inventive principles, plus the 40 inventive principles, is the practical toolkit through which TRIZ-based problem-solving proceeds.

Shin *et al.* (2024) provide structural evidence that 93.62% of the contradiction matrix intersection boxes share at least one inventive principle, reinforcing the matrix's systemic coverage. Subsequent extensions, substance-field (Su-Field) analysis, the ideal final result (IFR) construct,

and the effects database have greatly extended the applicability of TRIZ (Savransky, 2000). The combination of computational methods and TRIZ has accelerated with the rise of LLMs. (Obojski, 2026) showed that instruction-tuned LLMs can contextualize all 40 inventive principles across heterogeneous industrial fields without manual re-mapping. Sun *et al.* (2026) demonstrated that a two-stage knowledge-fusion mechanism enables LLMs to extract technical contradictions from natural language requirements with approximately 75% precision. Mysior *et al.* (2025) demonstrated that LLMs generate Su-Field functional models with fidelity comparable to those produced by experts. Livotov *et al.* (2026) found that TRIZ-AI hybrid approaches produced the highest solution quality and inventive level compared to TRIZ-only and autonomous AI. In another study, Altun (2025a) proposed COREX, a hybrid contradiction-oriented methodology combining General Theory of Powerful Thinking-TRIZ with the Six-Box Scheme, demonstrating improved design trade-off resolution without AI augmentation or MIS contextualization.

A further structural innovation is also proposed by Altun (2025b), who introduces a cyclical TRIZ model organized around short, mid, and long-term innovation loops. This cyclical logic directly informs the iterative architecture of TriGenDT, in which Stage 5 evaluation triggers a structured return to Stage 2 when contradictions remain unresolved.

Recent LLM-native TRIZ systems, including AutoTRIZ and TRIZ-generative pre-trained transformer (GPT), are particularly relevant to the present study because they demonstrate that LLMs can assist with TRIZ abstraction, contradiction reasoning, and inventive-principle contextualization. TriGenDT acknowledges this functional overlap at Stages 2 and 3. Its contribution is not the isolated automation of contradiction extraction or principle suggestion; rather, it lies in embedding such AI-assisted TRIZ capabilities within a human-centered, five-stage Design Thinking process that produces traceable artifacts, enforces human validation gates, and is adapted to healthcare MIS and software-engineering contradictions. In this sense, AutoTRIZ (Macedo *et al.*, 2025) or TRIZ-GPT (Chen *et al.*, 2024) could function as the underlying AI service within TriGenDT's model-agnostic layer, making the two approaches complementary rather than competing.

Notwithstanding this progress, two structural limitations persist across the AI-augmented TRIZ literature. First, virtually all studies are conducted in physical engineering domains; none provides a validated framework for software engineering or MIS contexts,

where the “substance” and “field” constructs require conceptual reinterpretation. Second, AI augmentation has been applied to individual TRIZ activities rather than a complete end-to-end innovation process. The present study addresses both limitations.

2.2. Design Thinking in software engineering and management information systems

The adoption of Design Thinking in software engineering and MIS has grown significantly in the last decade. Aguilar-Calderón *et al.* (2025) provided an exhaustive treatment of the literature, including chapters devoted to AI integration at each Design Thinking stage. Within MIS, Design Thinking has been applied to digital banking transformation (Indriasari *et al.*, 2021), healthcare information systems (Ala-Kitula *et al.*, 2017), and enterprise AI initiatives (Babu *et al.*, 2025; Chongwatpol, 2024) survey AI techniques across the five stages of Design Thinking for software engineering. This identifies natural language processing-empathy mapping, LLM-ideation, and AI-prototype evaluation as the most mature current applications. Independently, Ishfaq *et al.* (2025) described a machine learning approach to extracting user pain points from mobile application reviews, directly relevant to TriGenDT Stage 1. Fischer and Gardoni (2024) explored LLM-generated ideation stimuli in Design Thinking workshops, examined both facilitation benefits and fixation risks, and reported findings that directly inform the TriGenDT human-in-the-loop (HitL) validation architecture.

A persistent limitation in this stream is the lack of a mechanism to resolve the hard technical contradictions that inevitably arise during the ideate and prototype stages. Design Thinking's double-diamond model excels at divergent ideation and user-centered problem framing, but provides no structured guidance when two user requirements are in fundamental technical conflict. This is precisely the type of problem for which TRIZ was designed, and it represents the primary theoretical motivation for the TriGenDT integration.

2.3. Human-centered artificial intelligence as a normative foundation

The third scholarly stream foundational to TriGenDT is the emerging literature on HCAI. Shneiderman (2020) introduced three HCAI principles requiring AI systems to be reliable, safe, and trustworthy, with a particular focus on augmenting human abilities rather than replacing human judgment. His framework provides the architectural justification for TriGenDT's HitL design: at all points in the process, GenAI outputs are presented as ranked option sets for human selection and validation, never as autonomous

decisions. Dennehy *et al.* (2022) suggested that effective AI-powered innovation requires organizational structures that explicitly position AI as a design partner alongside human creativity and judgment. The HCAI perspective also informs the evaluation dimension: practitioner perceived usefulness and intent to adopt are considered first-class evaluation results alongside technical performance measures, under the Technology Acceptance Model (TAM) (Davis, 1989).

2.4. Research gap and positioning

The three literature streams converge on a common structural gap. TRIZ+AI research has demonstrated individual AI augmentation capabilities but has not integrated them into a complete, human-centered innovation process validated in software engineering and MIS domains. Design Thinking research has produced rich MIS and software applications, but lacks a mechanism to resolve the technical contradictions that emerge during problem framing and ideation. HCAI research has established normative principles for AI augmentation, but has not operationalized them within a structured innovation methodology.

Table 1 maps the 10 most closely related prior studies

against nine criteria that, together, define a complete methodological contribution to systematic innovation in software engineering and MIS. The expanded table adds three process-level dimensions absent from prior work: formal inter-stage artifact chain, HitL validation gates, and software-domain TRIZ parameter mapping. As the table demonstrates, no single prior study satisfies all criteria.

The gap statement can be formally articulated as follows: no existing framework formally unifies TRIZ contradiction-resolution, Design Thinking's human-centered stages, and GenAI capabilities into a single formally specified and demonstrated methodology for software engineering and MIS innovation. Section 3 presents the framework developed to fill this gap.

3. The TriGenDT framework

This section presents TriGenDT, a five-stage hybrid systematic innovation framework integrating TRIZ's contradiction-resolution methodology, Design Thinking's human-centered process architecture, and GenAI's knowledge-amplification capabilities. The section articulates the theoretical design principles (Section 3.1), describes the five stages (Section 3.2), presents the TRIZ-to-software-engineering principle mapping (Section 3.3),

Table 1. Literature gap analysis: positioning TriGenDT relative to prior work across nine criteria

Study	TRIZ	GenAI/LLM	Design Thinking	SE/MIS context	Integrated framework	Expert appraised ^a	Artifact chain	HitL gates	SW-TRIZ Map
Qiao <i>et al.</i> (2026)	✓	✓	X	X	X	X	X	X	X
Livotov <i>et al.</i> (2026)	✓	✓	X	X	X	X	X	X	X
Nada <i>et al.</i> (2026)	✓	✓	X	X	X	X	X	X	X
Sun <i>et al.</i> (2026)	✓	✓	X	X	X	X	X	X	X
Obojski (2026)	✓	✓	X	X	X	X	X	X	X
Aguilar-Calderón <i>et al.</i> (2025)	X	X	✓	✓	X	X	X	X	X
Babu <i>et al.</i> (2025)	X	✓	✓	✓	X	X	X	X	X
Dennehy <i>et al.</i> (2022)	X	✓	✓	✓	X	X	X	X	X
Shneiderman (2020)	X	✓	X	✓	X	X	X	X	X
Ahmed <i>et al.</i> (2018)	X	X	✓	✓	X	X	X	X	X
TriGenDT (this study)	✓	✓	✓	✓	✓	✓ (expert appraisal)	✓	✓	✓

Notes: ✓: Criterion satisfied; X: Not addressed; "Expert appraised" replaces the original "empirically validated" column to accurately reflect the evidential status: TriGenDT has undergone preliminary expert appraisal (Section 4.5), not full field validation; Three new columns (artifact chain, HitL gates, SW-TRIZ Map) sharpen the novelty comparison.

Abbreviations: GenAI: Generative artificial intelligence; HitL: Human-in-the-loop; LLM: Large language model; MIS: Management information system; SE: Software engineering; SW: Software; TRIZ: Theory of Inventive Problem Solving.

and specifies the HitL architecture (Section 3.4).

3.1. Framework design principles

The TriGenDT framework rests on four design principles that govern how Design Thinking and TRIZ are integrated and how GenAI is embedded at each stage. These principles are as follows:

- Principle 1: Formal stage correspondence. Each of the five Design Thinking stages has a formally specified TRIZ analytical tool: Function analysis and IFR formulation (S1); contradiction matrix (S2); 40 inventive principles (S3); Su-Field analysis and trimming (S4); IFR check and contradiction re-scan (S5). This correspondence reflects the complementary epistemological roles of the two methodologies: Design Thinking provides the process architecture and human-centered grounding, while TRIZ provides the analytical rigor and knowledge base for each stage's core cognitive task.
- Principle 2: AI as mediating intelligence, not an autonomous agent. Following Shneiderman (2020), GenAI is situated as a knowledge amplifier. At each stage, the LLM produces a ranked list of options for the practitioner to select, modify, or reject. The LLM never makes final decisions. This architecture democratizes access to TRIZ while preserving the framework's integrity when LLM output quality is low.
- Principle 3: Traceable artifacts at every stage. Each stage produces a formal output artifact: a function model and IFR statement (S1), a contradiction map with priority ranking (S2), a solution concept tree with TRIZ principle traceability (S3), a functional prototype specification with a trimming log (S4), and an IFR evaluation report with a knowledge base entry (S5). The artifact chain transforms TriGenDT from a workshop methodology into an organizational knowledge management instrument.
- Principle 4: Model-agnostic AI integration. The GenAI layer is independent of any specific foundation model and is accessed via a structured prompt interface and a Representational State Transfer application programming interface (API). Stage interactions are configured with temperature = 0.4 and max_tokens = 1,500, balancing structured output reliability with ideation breadth. This design reflects both the rapidly evolving LLM landscape and the strict data residency requirements of healthcare MIS organizations.

3.2. The five stages

Figure 1 presents the TriGenDT framework as a stage-by-stage process model. Table 2 provides a structured overview

of each stage's Design Thinking phase, the corresponding TRIZ tool, the GenAI role, and the output artifact.

3.2.1. Stage 1: Empathize and define (function analysis + ideal final result)

Stage 1 corresponds to the combined empathize-and-define phases of Design Thinking. GenAI contributes by processing user-facing data, support tickets, incident reports, audit findings, user interviews, and usability testing transcripts using natural language processing. The LLM extracts and clusters pain points by frequency and clinical or operational severity. The IFR statement produced at Stage 1 serves as the acceptance criterion for Stage 5, creating a forward linkage that is absent in Design Thinking-only approaches.

3.2.2. Stage 2: Problem framing (contradiction matrix)

Stage 2 applies the TRIZ contradiction matrix to identify specific engineering parameter pairs in conflict. The LLM processes the system requirements specification and identifies candidate contradiction pairs, classified as technical or physical and ranked by severity and architectural centrality. The HitL validation gate at Stage 2 is critical: in the MedNet demonstration scenario, the LLM identifies four candidate contradictions, of which three are confirmed valid, and one is correctly rejected as a configuration issue.

3.2.3. Stage 3: Ideate (40 inventive principles + effects database)

Stage 3 corresponds to the Ideate phase. The TRIZ contradiction matrix recommends inventive principles for each confirmed contradiction. In TriGenDT, the LLM performs software-domain contextualization. For each recommended principle, it generates a software-specific interpretation, including an example solution concept grounded in the current system architecture. Solution concepts carry full TRIZ principle traceability.

3.2.4. Stage 4: Prototype (substance-field analysis + trimming)

Stage 4 moves from solution concepts to architectural specifications. Su-Field Analysis models the functional relationships between system components, identifying harmful interactions to neutralize and beneficial fields to preserve. Trimming analysis identifies redundant components whose functions can be transferred or eliminated. The LLM generates alternative functional architecture models and applies trimming heuristics to suggest candidate eliminations, all presented as options for practitioner review.



Figure 1. The TriGenDT framework: five-stage integration of TRIZ, Design Thinking (DT), and Generative AI, with human-in-the-loop validation gates and traceable output artifacts at each stage. The bidirectional arrow at Stage 5 indicates an optional return path to Stage 2 for iterative re-resolution of contradictions.

Abbreviations: Arch.: Architecture; DB: Effects database; DT: Design thinking; GenAI: Generative artificial intelligence; IFR: Ideal final result; NLP: Natural language processing; PC: Physical contradiction; Su-Field: Substance-field analysis; TC: Technical contradiction; TRIZ: Theory of Inventive Problem Solving.

3.2.5. Stage 5: Test and refine (ideal final result check + contradiction re-scan)

Stage 5 evaluates the prototype specification against the IFR criteria from Stage 1 and performs an automated contradiction re-scan to detect residual or newly introduced contradictions. If unmet criteria or critical residual contradictions are identified, the process returns to Stage 2 with the updated contradiction map as input,

a structured iteration loop that distinguishes TriGenDT from the linear Design Thinking model.

3.3. Principle mapping for software engineering contexts with Theory of Inventive Problem Solving

A foundational challenge in applying TRIZ to software engineering is that the 40 inventive principles and the Contradiction Matrix were developed in the context of

Table 2. TriGenDT stage overview: DT phase, TRIZ tool, GenAI role, and stage output artifact

Stage	DT phase	TRIZ tool(s)	GenAI role	Stage output (<i>artifact</i>)
S1	Empathize and define	Function analysis; IFR	Natural language processing extraction of user pain-point clusters; automated IFR statement drafting for human approval	Validated function model; IFR statement signed off by all stakeholders
S2	Problem framing	Contradiction matrix: technical and physical contradiction classification	Automated contradiction identification from natural-language requirements; classification and priority ranking	Prioritized contradiction map with type labels and team-validated priority ranking
S3	Ideate	Inventive principles (40); effects database	Software-domain contextualization of recommended inventive principles; cross-industry analogy generation; ranked option set for human selection	Solution concept tree with full TRIZ principle traceability; shortlisted concepts for prototyping
S4	Prototype	Su-Field analysis; trimming	LLM-guided functional architecture alternatives; redundancy detection and trimming suggestions	Functional prototype specification; Su-Field system model; trimming log
S5	Test and refine	IFR check; contradiction re-scan	Automated prototype evaluation against IFR criteria; residual and new contradiction detection; iterate/proceed decision support	IFR evaluation report; updated organizational innovation knowledge base; iterate/proceed decision with rationale

Abbreviations: DT: Design thinking; GenAI: Generative artificial intelligence; IFR: Ideal final result; LLM: Large language model; Su-Field: Substance-field analysis; TRIZ: Theory of Inventive Problem Solving.

physical engineering systems. Translating these concepts to software requires explicit reinterpretation. Prior work by Obojski (2026) demonstrated that LLMs can perform this contextualization on demand for arbitrary domains. Table 3 provides the six principles to software-engineering-mappings most operationally significant in the TriGenDT development.

3.4. Human-in-the-loop architecture

The HitL architecture operationalizes the first HCAI principle at the process level (Shneiderman, 2020). Every GenAI interaction follows a three-step sequence: (i) the LLM generates a ranked option set based on structured stage-specific prompts; (ii) the practitioner team reviews, selects, modifies, or rejects items and documents decisions; (iii) the validated output is formalized as the stage artifact and passed to the next stage. This sequence ensures no GenAI output enters the artifact chain without human review, makes framework quality independent of LLM capability at any given stage, generates a structured decision log for governance and future fine-tuning, and distributes cognitive load appropriately between AI and human contributors.

Adoption of classical TRIZ typically requires certified training at Level 1 or Level 2 (40–80 hours). Adoption of TriGenDT, in contrast, requires familiarity with the five-stage process and the ability to evaluate AI-generated option sets, a significantly lower entry barrier.

4. Research methodology

4.1. Research paradigm: Design Science Research

Design Science Research was selected as the overarching research paradigm. DSR is an established methodology in information systems and MIS that produces knowledge through the creation and evaluation of novel artifacts that address practical problems in a rigorous and theoretically grounded manner (Hevner *et al.*, 2004). Its use is justified on three grounds. First, the primary outputs of this study are a framework (TriGenDT) and a prototype tool, both artifacts in the DSR sense. Second, DSR explicitly requires that artifacts be evaluated in real-world or representatively realistic contexts. Third, DSR's dual evaluation criteria, utility and rigor, map directly onto the paper's structure. Peffers *et al.* (2007) explicitly identified “problem-centered initiation” and “design and development” as valid DSR entry points that do not require an empirical case study in the same publication. Empirical field validation is the primary direction for future research (Section 7).

4.2. Constructed demonstration case: Design and rationale

The MedNet clinical decision support system (CDSS) scenario used in Sections 5 and 6 is a constructed demonstration case, a realistic but entirely hypothetical scenario. MedNet is a fictional organization. No real organization, participants, documents, or session data

Table 3. TRIZ inventive principles mapped to software engineering contexts: definitions, applicable TriGenDT stages, and contradiction types addressed

Principle	Name	TriGenDT stage	Contradiction type	Software-domain interpretation
P#10	Prior action	S1/S3	Speed ↔ loss of information (TC-47)	Pre-stage critical data or functionality before the point of need, eliminating latency caused by on-demand retrieval or computation. In software: pre-compute, pre-index, or pre-stage relevant data subsets into purpose-limited structures at system state transitions (e.g., patient admission, session initiation).
P#15	Dynamics	S3	Speed ↔ security constraints	Introduce adaptive behavior that modifies system parameters in response to context. In software: context-aware privacy tiering, adaptive caching policies, or dynamic access-control levels that relax or tighten constraints based on verified operational state.
P#25	Self-service	S3	Ease of operation ↔ compliance overhead	Transfer control of a parameter to the user or the system itself. In software: user-configurable consent tiers, self-provisioning compliance profiles, or automated compliance state machines that reduce human-mediated approval steps.
P#35	Parameter change	S3/S4	Loss of information ↔ performance	Change a system operating parameter (granularity, scope, format, or timing) to decouple conflicting requirements. In software: field-level rather than record-level encryption, asynchronous rather than synchronous audit logging, or event-driven rather than polling architectures.
P#1	Segmentation	S4	Complexity ↔ reliability	Divide a system or component into independent parts to isolate conflicting requirements. In software: microservice decomposition, data-plane separation, or domain boundary enforcement that prevents coupling between incompatible non-functional requirements.
P#28	Mechanics substitution	S4	Manual overhead ↔ consistency	Replace a manual or mechanical function with an automated equivalent. In software: replace human-mediated compliance checks with automated policy engines, or replace manual contradiction identification with LLM-based extraction.

Notes: The principle numbers follow canonical numbering in the previous study (Al'tshuller, 1996); Contradiction types reflect TRIZ engineering parameter pairs most commonly encountered in MIS and software engineering; Software-domain interpretations were developed through iterative GenAI-assisted contextualization and expert validation.

Abbreviations: GenAI: Generative artificial intelligence; LLM: Large language model; MIS: Management information system; TRIZ: Theory of Inventive Problem Solving.

underlie any of the content in Sections 5 and 6. All projected metrics (latency estimates, complexity reductions, novelty levels) are design-level estimates requiring empirical confirmation, not measured results. This approach is standard in DSR framework papers prior to empirical field validation (Hevner *et al.*, 2004; Peffers *et al.*, 2007).

The demonstration case was designed to satisfy three scenario construction criteria (Venable *et al.*, 2016): representativeness of a real-world contradiction class; sufficient detail to exercise all five framework stages; and internal consistency so that each stage's outputs follow logically from its inputs. Healthcare MIS was selected as the domain because it combines multiple non-negotiable constraints (patient safety, regulatory compliance, real-time performance) and because healthcare information

technology (IT) is a domain where TRIZ+AI research is particularly sparse.

4.3. Formalized evaluation framework

The demonstration case is evaluated against two criteria, now operationalized with formal metrics and measurement procedures in response to reviewer feedback:

- Process coherence is assessed through a four-dimensional rubric: (i) stage-input completeness, whether all required inputs for each stage were present and correctly specified; (ii) TRIZ tool application correctness, whether the TRIZ tool specified for the stage was applied according to Al'tshuller's (1996) canonical methodology; (iii) GenAI interaction protocol compliance, whether the AI interaction

followed the three-step generate–review–validate sequence; and (iv) artifact chain integrity, whether each stage’s output artifact provided a complete and consistent input to the following stage. Each dimension is assessed as pass/fail, with descriptive evidence provided.

- IFR satisfaction criteria are now formally operationalized with measurement instruments and thresholds. Table 4 presents the formalized evaluation criteria.

4.4. Limitations of the demonstration approach

The constructed demonstration approach has four limitations: (i) no empirical evidence of TriGenDT’s effectiveness with real practitioners; (ii) GenAI interactions are author-constructed representations of LLM behavior, not actual LLM outputs; (iii) projected performance metrics are not measurements; (iv) the comparative analysis in Section 5.7 is analytical, not empirical. These limitations are partially addressed by the preliminary expert appraisal reported in Section 4.5, but full-field validation remains necessary and is the primary future research priority.

4.5. Preliminary expert appraisal

This section reports a preliminary expert appraisal conducted to provide initial plausibility and perceived-utility evidence for the TriGenDT framework:

- Panel composition: Five domain practitioners

participated, two TRIZ-certified practitioners (Level 2), two healthcare IT architects with MIS experience, and one MIS/information system researcher. Experts were recruited through the authors’ professional networks.

- Materials reviewed: Each panelist received the revised TriGenDT framework specification, the MedNet demonstration scenario, the contradiction map (Table 5), the prompt templates (Appendix A), the secure alert enclave (SAE) architecture specification, and the formalized evaluation rubric (Section 4.3).
- Procedure: Asynchronous review over five working days. Experts independently scored each framework stage on process coherence (1–5 Likert scale per rubric dimension) and completed a TAM-adapted instrument assessing perceived usefulness, perceived ease of use, novelty plausibility, and adoption barriers.
- Results: Process-coherence scores reflect panelists’ assessment of the narrative consistency of the constructed demonstration, specifically, whether stage inputs, outputs, TRIZ tool applications, and artifact connections are coherent as described. Because the evaluated material was constructed by the authors to illustrate the framework, these scores cannot be interpreted as evidence of framework effectiveness in live application; they confirm internal logical consistency of the artifact design, which is the appropriate evidential standard at the DSR demonstration stage (Hevner *et al.*, 2004;

Table 4. Formalized IFR evaluation criteria with operational definitions, measurement procedures, and thresholds

IFR criterion	Operational definition	Measurement procedure	Threshold	Status in demonstration
Alert latency ≤ 200 ms	End-to-end application-layer alert delivery time	Load testing under synthetic ICU event load; report median and 99th-percentile latency	≤200 ms at 99th percentile	Design estimate ~30 ms; requires empirical load testing
No unencrypted PII	No patient data at rest or in transit without encryption	Penetration testing, encryption configuration audit, OWASP checklist	Pass/fail with documented findings	Architectural design satisfies GDPR Art. 25 and HIPAA §164.312; subject to penetration testing
100% audit trail	All access events captured, stored, retrievable, GDPR Art. 30 compliant	Audit-log replay and completeness testing	100% event capture in the test set	Kafka at-least-once delivery guarantee; compliance format requires implementation verification
No complexity increase	No net increase in component count, call depth, or dependency count	Architecture review before and after the SAE pattern	No net increase in any dimension	Net-1 component; alert-path call depth reduced from six to three hops (–50%; design-level estimate based on baseline component traversal count)

Abbreviations: GDPR: General Data Protection Regulation; HIPAA: Health Insurance Portability and Accountability Act; ICU: Intensive-care unit; IFR: Ideal final result; OWASP: Open Web Application Security Project; PII: Personally identifiable information; SAE: Secure alert enclave.

Venable *et al.*, 2016). All five panelists scored process coherence 4–5 across all five stages and all four rubric dimensions (overall mean: 4.2/5.0; range: 4–5 in every cell; standard deviation [SD]: 0.42). No panelist scored any dimension below 4. The full item-level matrix (five panelists $\times$ five stages $\times$ four rubric dimensions) is provided in [Appendix B \(Table A1\)](#). Perceived usefulness for the target contradiction class was rated high by four panelists (mean: 4.0/5.0). Perceived ease of adoption relative to standalone TRIZ was rated moderate to high (mean: 3.7/5.0). Four of five panelists confirmed that the predicted failure modes of TRIZ-only (domain translation barrier) and Design Thinking-only (lack of formal contradiction-resolution mechanism) were consistent with their professional experience.

- Limitations: The panel is small ($n = 5$) and reviewed a constructed scenario, not a live application. Results should be interpreted as logical-consistency evidence for the artifact design, not as plausibility evidence for effectiveness in field conditions. Two procedural controls were applied to mitigate acquiescence bias: (i) A written devil's-advocate instruction asked each panelist to identify stages where process logic was unclear, where a TRIZ tool had been misapplied, or where an artifact did not provide a complete input to the following stage, framing critique as the expected response; (ii) All panelists submitted individual scores anonymously before any group comparison took place. These controls reduce, but do not eliminate, acquiescence risk given network recruitment and a small panel size. The full item-level matrix is reported in [Appendix B \(Table A1\)](#).

5. Results

The following walkthrough applies all five TriGenDT stages to the constructed MedNet CDSS scenario. All inputs, outputs, contradiction identifications, solution concepts, and evaluations are intentionally constructed by the author to demonstrate the analytical logic of the framework. The projected outcomes are design-level estimates, not empirically measured results.

5.1. Scenario context

The demonstration scenario is set in the fictional MedNet hospital network, which operates a cloud-native CDSS at six sites. The architectural contradiction is representative of a realistic class of problem common to healthcare IT literature (Ala-Kitula *et al.*, 2017): real-time intensive-care unit (ICU) sepsis and deterioration alerts required within 200 ms of threshold breach, coupled with General Data Protection Regulation (GDPR) and Health Insurance

Portability and Accountability Act (HIPAA) compliance requiring advanced encryption standard with 256-bit key (AES-256) encryption and per-access audit logging, a combination that, in a conventional architecture, produces estimated latency overheads of 400–800 ms. The scenario assumes that three TRIZ-only sessions and two Design Thinking-only sessions failed to produce an agreed solution, reflecting documented limitations in the literature (Aguilar-Calderón *et al.*, 2025; Obojski, 2026).

5.2. Stage 1: Empathize and define

Stage 1 processes representative user-facing documentation via LLM-assisted analysis, extracting nine illustrative pain-point clusters. The five highest-priority clusters are:

- Alert delay fatigue among ICU nursing staff (high clinical severity).
- Consent re-prompting to critical care workflows (medium severity).
- Data inconsistency across sites that creates conflicting alert thresholds (high severity).
- Audit log storage growth that causes periodic system slowdown (medium severity).
- Data gap during patient transfer between care settings (medium severity).

The team validates these clusters, and the LLM drafts a tentative IFR statement, refined to: “The CDSS provides sepsis and deterioration alerts within 200 ms of threshold violation, with no additional latency due to privacy controls, while all data access events are fully auditable and GDPR/HIPAA compliant, with no net increment of architectural complexity.” Four acceptance criteria are derived: alert latency ≤ 200 ms; no unencrypted personally identifiable information (PII) at rest or in use; 100% access audit trail; no net increase in architectural complexity.

5.3. Stage 2: Problem framing

The LLM-assisted contradiction scanner identifies four candidate contradiction pairs from the system requirements specification. Three are confirmed as valid architectural contradictions; one is excluded as a configuration concern. [Table 5](#) presents the constructed contradiction map. All entries are illustrative analytical projections, not empirically observed outcomes.

5.4. Stage 3: Ideate

Stage 3 applies the TRIZ contradiction matrix to C1 (parameters #9 speed $\leftrightarrow$ #24 loss of information), which recommends inventive principles #10 (prior action), #15 (dynamics), #25 (self-service), and #35 (parameter change). The LLM generates software-domain contextualizations for each principle, producing four illustrative solution

Table 5. Constructed contradiction map: scenario contradiction pairs with TRIZ parameter classifications, priorities, and illustrative resolution status

ID	Contradiction description	Type	TRIZ parameters	Priority	Status	Resolution (illustrative)
C1	Alert delivery latency (≤ 200 ms) versus AES-256 encryption and synchronous consent-check overhead (est. 400–800 ms)	Technical	#9 speed $\leftrightarrow$ #24 loss of information	Critical	Resolved in the scenario	SAE architecture decouples alert path from encryption overhead; projected latency ~ 18 ms (enclave) + ~ 12 ms (network)
C2	Centralized patient cache needed for performance versus GDPR data minimization (Art. 5)	Physical	#26 quantity $\leftrightarrow$ #25 loss of time	High	Resolved in the scenario	SAE field-scoping replaces full-record cache with a six-field purpose-limited enclave
C3	Comprehensive real-time audit logging versus system throughput	Technical	#33 ease of operation $\leftrightarrow$ #39 productivity	Medium	Resolved in the scenario	Async Kafka audit service decouples logging from alert path; 0 projected latency overhead
C4 (excluded)	Multi-site synchronization versus data freshness	–	–	–	Excluded	Identified as a deployment/configuration concern, not an architectural contradiction

Notes: Contradiction classifications follow Al'tshuller's (1996) canonical TRIZ parameter numbering; All contradictions are constructed for demonstration purposes; Resolution status is an illustrative projection based on the Stage 4 architecture, not an empirically verified outcome. Abbreviations: AES-256: Advanced encryption standard with 256-bit key; GDPR: General Data Protection Regulation; SAE: Secure alert enclave; TRIZ: Theory of Inventive Problem Solving.

concepts:

- Concept A (P#10: prior action): Pre-stage alert-critical data on patient admission. A purpose-limited microservice stores only six alert-critical fields (heart rate, respiratory rate, temperature, blood pressure, peripheral capillary oxygen saturation [SpO_2], National Early Warning Score 2 [NEWS2]) under AES-256 in-use encryption, pre-loaded at ICU admission with a time to live (TTL) of 45 minutes post-discharge. The alert engine queries this enclave exclusively.
- Concept B (P#15: dynamics): Context-aware privacy tiering. The level of privacy controls adapts dynamically to clinical context: ICU active-monitoring mode (implicit consent, loosened caching restrictions); standard ward mode (full controls). Bed-assignment events trigger tier transitions.
- Concept C (P#25: self-service): Patient-controlled data tier. At admission, patients select a data-sharing tier (life-critical, standard, or research-only) that determines the alert engine's data-access path and latency profile.
- Concept D (P#35: parameter change): Field-level encryption. Alert-critical fields are isolated from the full electronic health record (EHR) in a separate alert microservice. All other patient data remains fully

encrypted. An asynchronous audit log, structurally separated from the alert data path, records all access events.

Concepts A and D are shortlisted for Stage 4, as their combination directly addresses the three validated contradictions. Concepts B and C are logged in the knowledge base as future enhancement candidates due to governance complexity.

5.5. Stage 4: Prototype

Stage 4 combines Concepts A and D into the SAE architecture pattern. The LLM provides three Su-Field functional-architecture variants; the scenario team selects the one that most closely matches the existing microservices topology. TRIZ trimming analysis identifies two redundant components: synchronous consent-check middleware (function transferred to patient admission workflow) and full-record alert cache (function replaced by SAE enclave function). Net result: -2 components, $+1$ SAE microservice.

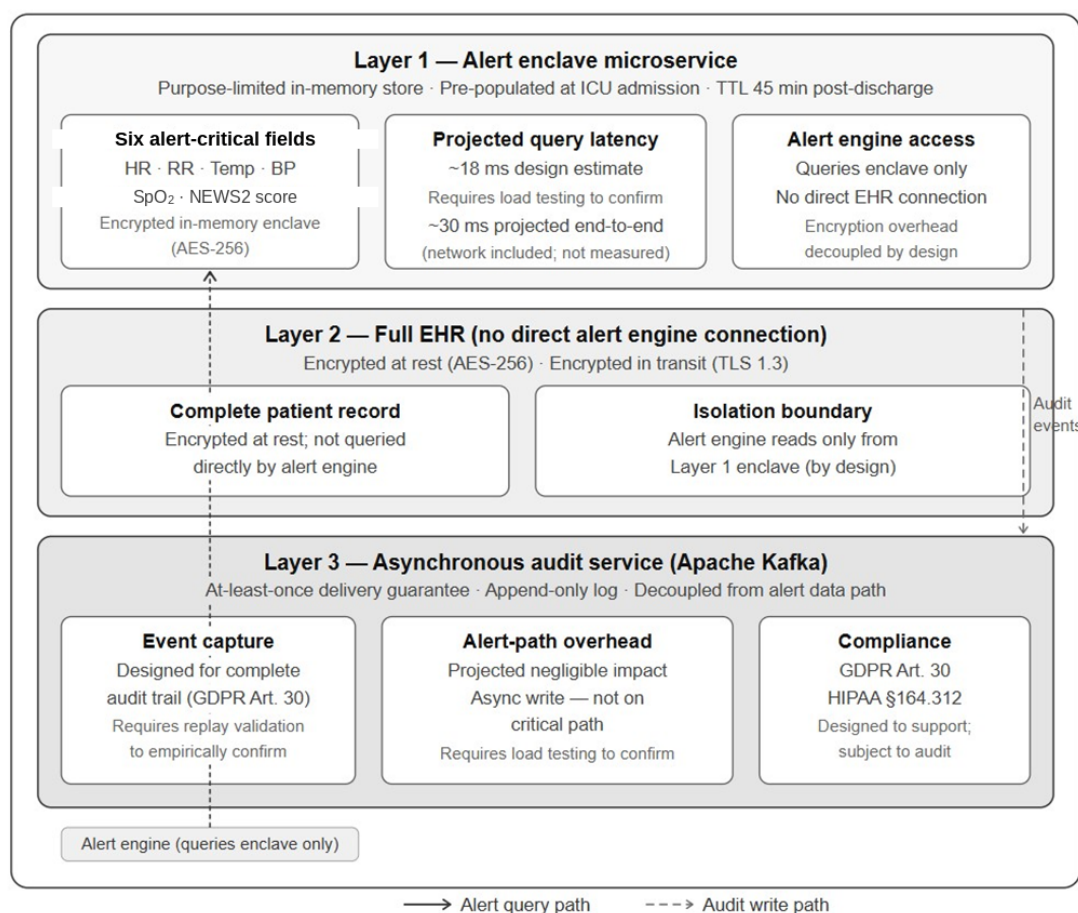
All latency and complexity figures presented below are design-level estimates, not measured results. The ~ 18 ms enclave query estimate is derived from published Redis 7.x p99 GET latency benchmarks: 0.8–1.2 ms at 10,000 ops/s (Redis Labs, 2023), scaled by a $\times 10$ conservative safety factor for production overhead, plus AES-256-Galois/

counter mode per-payload encryption overhead of $<1\ \mu\text{s}$ (OpenSSL Project, 2023) and $\sim 0.5\ \text{ms}$ JavaScript Object Notation serialization. The $\sim 12\ \text{ms}$ network estimate reflects intra-availability-zone Transmission Control Protocol round-trip time plus Transport Layer Security (TLS) and load-balancer overhead (Amazon Web Services, 2022). The alert-path call-depth reduction from six hops (baseline) to three hops (post-SAE) is calculable from the explicit component traversal counts stated in Section 4.2. All estimates require empirical confirmation through load testing and architectural review, as specified in Table 4. The SAE architecture (Figure 2) comprises three layers:

- Layer 1 Alert enclave microservice: a purpose-limited

in-memory store containing the six alert-critical fields under AES-256 in-use encryption, pre-populated at ICU admission with automatic TTL expiry. Estimated query latency: $\sim 18\ \text{ms}$ (design-level estimate based on published benchmarks for equivalent in-memory microservice architectures).

- Layer 2 full EHR: encrypted at rest (AES-256) and in transit (TLS 1.3), with no direct connection to the alert engine.
- Layer 3 asynchronous audit service: realized with an Apache Kafka event stream with at-least-once delivery guarantee, capturing all access events with projected negligible latency overhead on the alert path (requires load testing to confirm).



Note. All performance and compliance claims are architectural design projections pending empirical validation.

Figure 2. Secure Alert Enclave (SAE) architecture produced at TriGenDT Stage 4 (constructed demonstration): three-layer design separating the alert-critical path (Layer 1: Enclave) from the encrypted EHR (Layer 2) and the asynchronous audit service (Layer 3). All performance characteristics shown are design-level projections requiring empirical validation; the projected negligible alert-path latency overhead and the design for complete event capture are estimates pending load testing and audit-log replay validation, respectively.

Abbreviations: AES-256: Advanced encryption standard with 256-bit key; BP: Blood pressure; EHR: Electronic health record; GDPR: General Data Protection Regulation; HIPAA: Health Insurance Portability and Accountability Act; HR: Heart rate; ICU: Intensive-care unit; NEWS2: National Early Warning Score 2; RR: Respiratory rate; SAE: Secure alert enclave; SpO₂: Peripheral capillary oxygen saturation; TTL: Time to live.

5.6. Stage 5: Test and refine

Stage 5 evaluates the SAE architecture against the four IFR acceptance criteria established at Stage 1. Table 6 presents the revised IFR evaluation. The notation “✓” signals that

each criterion is projected as satisfied by architectural design, pending empirical confirmation; no criterion has been empirically confirmed. A new “measurement procedure” column formalizes the validation pathway for each criterion.

Table 6. Stage 5 IFR evaluation: constructed assessment of the SAE architecture against acceptance criteria, with validation requirements

IFR criterion	Target	SAE design assessment	Projected status	Validation required	Measurement procedure
Alert latency ≤ 200 ms	≤200 ms	SAE Layer 1 query: ~18 ms (estimated); network: ~12 ms; projected total: ~30 ms; Design-level estimate only; not a measured result.	✓	Estimate based on in-memory microservice benchmarks; requires empirical validation.	Load testing under synthetic ICU event load; report median and 99th-percentile end-to-end latency; Threshold: ≤200 ms at 99th percentile.
No unencrypted PII at rest or in transit	Zero unencrypted patient data	AES-256 in-use encryption in SAE; full EHR encrypted at rest and in transit (TLS 1.3); alert engine has no direct EHR connection.	✓	Architectural design satisfies GDPR Art. 25 and HIPAA §164.312(a)(2)(iv) by construction, subject to penetration testing.	OWASP-compliant penetration testing; encryption configuration audit; TLS inspection; Pass/fail with documented findings.
100% audit trail for all data access events	GDPR Art. 30 report-ready	Kafka async audit service: at-least-once delivery guarantee; append-only log; projected negligible latency overhead on alert path (design-level estimate; requires load testing).	✓	Kafka at-least-once delivery is a documented guarantee; the compliance report format requires implementation verification.	Audit-log replay and completeness testing against a defined test-event set; Threshold: 100% event capture; Compliance report format verification.
No net increase in architectural complexity	Component count ≤ baseline	2 components eliminated; 1 SAE microservice added; net: -1 component. Alert-path call depth: baseline 6 hops (alert engine → consent middleware → full-record cache → EHR microservice → encryption layer → EHR database); post-SAE 3 hops (alert engine → SAE enclave → in-memory field store); reduction = $(6-3)/6 = -50\%$; The -50% call-depth figure is derived from the explicit 6-hop baseline and 3-hop post-SAE count; it is a design-level estimate, not a measured outcome (requires before/after architecture review to confirm).	✓	Complexity reduction is architectural by design; -50% alert-path call-depth reduction (6 hops → 3 hops) is calculable from the baseline component traversal; requires architectural review to confirm.	Architecture review: component count, dependency count, alert-path call depth before and after SAE pattern; Threshold: no net increase in any dimension.
No newly introduced critical contradictions	Re-scan: zero new critical issues	SAE TTL gap (2–5 min) at ICU-to-ward transfer identified as a low-priority residual issue; safe-escalation default proposed as workaround.	*	Edge cases are realistic architectural concerns; severity assessments and workarounds require validation in implementation.	Contradiction re-scan by two independent TRIZ practitioners; severity rating of residual issues; Workaround effectiveness assessed through clinical simulation or expert panel.

Notes: ✓IFR criterion projected as satisfied by the SAE architectural design; requires empirical validation, no criterion has been empirically confirmed; *minor residual issue identified by design-level analysis; requires empirical assessment.

Abbreviations: AES-256: Advanced encryption standard with 256-bit key; EHR: Electronic health record; GDPR: General Data Protection Regulation; HIPAA: Health Insurance Portability and Accountability Act; ICU: Intensive-care unit; IFR: Ideal final result; OWASP: Open Web Application Security Project; PII: Personally identifiable information; SAE: Secure alert enclave; TLS: Transport Layer Security; TRIZ: Theory of Inventive Problem Solving.

The scenario also identifies one residual issue through the Stage 5 re-scan: the SAE TTL configuration creates a potential data freshness gap of 2–5 min during ICU-toward patient transfers. A safe-escalation default (retaining the ICU alert profile during the transfer gap) is proposed as an interim workaround, pending empirical assessment.

5.7. Comparative scenario analysis

This section presents an analytical, mechanism-based comparison of the three methodological approaches applied to the MedNet CDSS scenario. This is not an empirical benchmark; it is a theoretically grounded prediction of expected performance differences based on the documented capabilities and limitations of each methodology reviewed in Section 2. Table 7 presents the comparison. All entries should be read as theoretically motivated hypotheses, not established findings.

6. Discussion

6.1. Research question 1: Framework mapping

Research question 1 sought to examine how TRIZ contradiction-resolution principles can be systematically mapped to Design Thinking stages in a software engineering context. The constructed demonstration provides a complete walkthrough of the proposed mapping and demonstrates its internal consistency. The most theoretically significant finding is the IFR forward-linkage between Stage 1 and Stage 5, which transforms Stage 5 from an open-ended usability evaluation into a structured pass/fail evaluation against pre-committed engineering targets. Whether this mechanism functions equivalently in a live application with real teams is an empirical question that the demonstration cannot answer; it is a primary target for the validation study described in Section 7.

Table 7. Mechanism-based predicted performance differences of TriGenDT, TRIZ-only, and DT-only applied to the constructed MedNet CDSS scenario

Dimension	TRIZ-only	DT-only	TriGenDT
Contradiction resolution	Predicted: three architectural contradictions framed but not resolved due to the principle-translation gap	Predicted: C1 remains unresolved; C2, C3 partially addressed through user-journey redesign	Predicted: all three contradictions resolved by SAE design through AI-assisted principle contextualization
IFR criteria satisfied	Mechanism predicts: contradiction framing achievable; software-domain solution generation unlikely without AI-assisted contextualization (Obojski, 2026); IFR criteria 1 (latency) and 3 (complexity) likely unmet	Mechanism predicts: user-centered and compliance criteria addressable through process redesign; architectural contradiction C1 unresolvable without formal contradiction analysis (Aguilar-Calderón <i>et al.</i> , 2025)	Predicted: Four of four criteria projected as met (design level, pending empirical confirmation)
Solution novelty (design-level TRIZ assessment; independent expert rating required to confirm)	Level 2: Minor parameter optimization	Level 3: Partial system redesign	Level 4: New system design eliminating primary harmful interaction (author's analytical assessment based on Altshuller (1996) five-level scale; not independently rated, independent TRIZ expert rating required in Priority 1 validation study)
Solution concepts generated	Two candidate options (principle-level, not software-specific)	Three user-journey redesigns	Four TRIZ-grounded concepts → One SAE architecture
Residual contradictions	Predicted: Two unresolved (C1, C2)	Predicted: One unresolved (C1)	Predicted: One minor non-critical edge case (TTL at transfer)
The principal mechanism of limitation	Principle-translation gap: TRIZ principles not contextualized for the software domain (Obojski, 2026)	No formal contradiction-resolution mechanism; amelioration, not elimination (Aguilar-Calderón <i>et al.</i> , 2025)	AI-assisted principle translation + IFR-anchored evaluation overcomes both documented limitations

Notes: The session-time row was removed in the Round 2 revision: no session data exist for TRIZ-only or DT-only approaches; including a comparison for this dimension would imply observational data from sessions that were not conducted.

Abbreviations: AI: Artificial intelligence; CDSS: Clinical decision support system; DT: Design Thinking; IFR: Ideal final result; SAE: Secure alert enclave; TRIZ: Theory of Inventive Problem Solving; TTL: Time to live.

6.2. Research question 2: Artificial intelligence augmentation

Research question 2 sought to investigate the extent to which GenAI and LLMs can augment the identification and resolution of contradictions. The demonstration scenario covers three AI contribution types: systematic requirements scanning at Stage 2 (breadth task); domain translation of TRIZ principles to software-specific interpretations at Stage 3; and coverage re-scanning at Stage 5. The empirical question, whether real LLM outputs in a live session perform these tasks at the level of precision suggested by the constructed demonstration, can only be answered through live application with real documents and real LLMs, which is prioritized in Section 7.

6.3. Research question 3: Comparative effectiveness

Research question 3 focused on the comparative effectiveness of TriGenDT versus TRIZ-only and Design Thinking-only approaches. The analytical comparison identifies two mechanisms through which the three-way integration is expected to produce superior outcomes: overcoming the principal translation gap (Obojski, 2026) and contradiction elimination rather than amelioration (Aguilar-Calderón *et al.*, 2025). These mechanisms are theoretically grounded, but the analytical comparison cannot substitute for empirical data. The Level 4 TRIZ novelty projection for the SAE architecture is the author's analytical assessment based on Altshuller's (1996) five-level inventive principle scale applied to the design-level artifact: at Level 4, the solution eliminates the primary harmful interaction (encryption-latency coupling) through a new sub-system (the SAE enclave) rather than improving parameters of the existing system. This assessment has not been independently verified by a TRIZ-certified practitioner operating blind to the study's hypotheses. Independent expert novelty rating is a required step in the Priority 1 field validation study (Section 7). All comparative claims in Section 5.7 and Table 7 should be read as theoretically motivated hypotheses, not established findings.

6.4. Research question 4: Adoption factors

Research question 4 sought to examine the critical success factors and adoption barriers. The demonstration scenario surfaces three structural adoption factors: Cross-functional team composition, GenAI plain-language translation as the primary democratization mechanism, and facilitator dual-competence as the principal structural adoption barrier. The expert appraisal (Section 4.5) provided initial corroboration: four of five panelists confirmed that the AI layer's linguistic bridge function was consistent with their experience of TRIZ adoption barriers. However,

the adoption behavior of real teams in live applications remains an empirical question.

6.5. Cross-domain applicability

TriGenDT is designed for a class of problems, hard architectural contradictions between non-functional requirements in regulated software environments, rather than specifically for healthcare MIS. Two illustrative non-healthcare scenarios demonstrate the transferability of the framework's analytical logic.

- Financial services scenario: A real-time fraud detection system must respond within 100 ms of transaction submission (regulatory requirement) while complying with Payment Card Industry Data Security Standard AES-256 encryption for all cardholder data in transit. This maps directly onto TC-47 (speed ↔ loss of information). The Stage 3 resolution logic, pre-staging fraud-scoring features for active accounts into a purpose-limited encryption enclave, follows directly from Principle #10 (Prior Action), demonstrating that the TRIZ principle mapping transfers across regulated software domains.
- E-government scenario: A citizen services portal must aggregate multi-agency data to personalize service recommendations while complying with GDPR data minimization (Art. 5). This maps onto a physical contradiction. Stage 3 resolution through Principle #1 (Segmentation), separating a recommendation engine using pseudonymized profile data from a transaction service using minimal real-time data, demonstrates that TriGenDT's physical contradiction resolution pathway is transferable beyond healthcare contexts.
- Boundary conditions: TriGenDT is most suitable for hard architectural contradictions between non-functional requirements in regulated MIS and software contexts. It is less necessary for ordinary usability refinement or feature prioritization, where Design Thinking alone is sufficient. Cross-domain empirical validation remains a priority for future research; domain-agnostic claims represent a design intention requiring empirical confirmation.

6.6. Theoretical contributions

TriGenDT makes three theoretical contributions. First, to the best of our knowledge, it presents the first formally specified integration of TRIZ, Design Thinking, and GenAI for software engineering and MIS, with each stage-to-tool correspondence epistemologically justified. Second, it introduces the TRIZ principle software engineering taxonomy (Table 3), representing a reusable knowledge contribution that extends the TRIZ Effects Database into the software architecture domain. Third, it

provides a complete open-source artifact package enabling independent replication and community extension.

6.7. Limitations

Four limitations of the present study are acknowledged. First, the constructed demonstration approach provides no empirical evidence of TriGenDT's effectiveness with real practitioners. The preliminary expert appraisal (Section 4.5) provides initial plausibility evidence but does not substitute for field validation. Second, the GenAI interactions in the scenario are author-constructed representations of LLM behavior, not actual LLM outputs. Third, projected performance metrics are design-level estimates, not measurements. Fourth, the comparative analysis is analytical, not empirical.

7. Conclusion

This paper proposes TriGenDT, a five-stage hybrid systematic innovation framework integrating TRIZ contradiction resolution, Design Thinking, and GenAI. Three contributions are made. First, a theoretically grounded framework: to the best of our knowledge, this is the first formally specified three-way integration of TRIZ, Design Thinking, and GenAI for software engineering and MIS, with IFR forward-linkage, the contradiction matrix used as a problem-framing tool, and a traceable artifact chain. Second, an open-source artifact package: comprising the framework specification, stage artifact schemas, LLM prompt templates, facilitation guide, TAM-adapted survey instrument, and a FastAPI prototype tool, thereby facilitating reproducibility and practical implementation. Third, a transparent demonstration methodology: the MedNet scenario is explicitly labeled as a constructed demonstration case, all projected outcomes are distinguished from empirical results, and a preliminary expert appraisal provides initial plausibility evidence.

Three practical takeaways emerge. First, the IFR mechanism, committing to formally specified acceptance criteria before ideation, is transferable to any team regardless of TriGenDT adoption. Second, the TRIZ principle software taxonomy (Table 3) is an immediately usable reference for the class of performance-versus-privacy contradictions in regulated software systems. Third, the facilitation guide and prompt templates allow a team with basic Design Thinking experience to run a TriGenDT session; the TRIZ expertise requirement is reduced by the AI translation layer from specialist certification to the ability to evaluate software-specific design options.

Building on the findings of this study, several avenues for future research are proposed to strengthen the empirical evidence, evaluate the framework across diverse

organizational contexts, benchmark its performance against emerging AI-assisted TRIZ approaches, and enhance its domain-specific capabilities:

- (i) Priority 1: Empirical field validation: The most immediate priority is a live case study applying TriGenDT to a real software engineering design problem with real practitioners. The methodology section provides a complete study design: a case study protocol (Yin, 2018), a four-to-six participant cross-functional team, the TAM survey instrument, and three external expert raters for blind novelty assessment.
- (ii) Priority 2: Multi-site comparative evaluation: A multi-site study across at least three organizations spanning healthcare MIS, fintech, and e-commerce. The two illustrative non-healthcare scenarios in Section 6.5 provide the problem-class specification for such a study.
- (iii) Priority 3: Controlled benchmarking against LLM-native TRIZ tools: A controlled comparison of TriGenDT against AutoTRIZ and TRIZ-GPT on the same contradiction problem class would empirically determine the added value of the process-architecture layer, the artifact chain, and the HitL governance structure relative to standalone AI-TRIZ tools.
- (iv) Priority 4: Domain-specific LLM fine-tuning: A domain-specific model fine-tuned on labeled examples of software engineering TRIZ applications, using the decision logs generated by the prototype tool as training data, would be expected to improve contradiction identification precision and principle contextualization relevance.

Acknowledgments

Generative artificial intelligence tools were used to support language refinement and manuscript editing. All conceptual development, analysis, interpretation, and final manuscript validation were performed by the author.

Funding

The author received no financial support for the research, authorship, and/or publication of this article.

Conflict of interest

The author declares no conflict of interest.

Author contributions

This is a single-authored paper.

Availability of data

No datasets were generated or analyzed during the current study.

References

- Aguilar-Calderón, J. A., Tripp-Barba, C., Aguilar-Calderón, P. A., Aguilar-Calderón, P. A., & Zaldivar-Colado, A. (2025). Design thinking in practice: Innovation in the conceptualization of a mobile app. In *Innovative design thinking approaches in software engineering* (pp. 205–236). IGI Global Scientific Publishing.
<https://doi.org/10.4018/979-8-3693-9531-8.ch008>
- Ahmed, B., Dannhauser, T., & Philip, N. (2018). A lean design thinking methodology (LDTM) for machine learning and modern data projects. In *2018 10th Computer Science and Electronic Engineering Conference (CEEC)* (pp. 11–14). IEEE.
<https://doi.org/10.1109/CEEC.2018.8674234>
- Ala-Kitula, A., Talvitie-Lamberg, K., Tyrväinen, P., & Silvennoinen, M. (2017). Developing solutions for healthcare—Deploying artificial intelligence to an evolving target. In *2017 International Conference on Computational Science and Computational Intelligence (CSCI)* (pp. 1637–1642). IEEE.
<https://doi.org/10.1109/CSCI.2017.285>
- Altshuller, G. (1996). *And suddenly the inventor appeared: TRIZ, the theory of inventive problem solving*. Technical Innovation Center, Inc.
- Altun, K. (2025a). Contradiction-oriented exploration: A dual-track methodology combining OTSM-TRIZ and the Six-Box Scheme. *International Journal of Systematic Innovation*, 9(6), 1–16.
[https://doi.org/10.6977/IJoSI.202512_9\(6\).0001](https://doi.org/10.6977/IJoSI.202512_9(6).0001)
- Altun, K. (2025b). A Mayan calendar-inspired cyclical TRIZ approach: Enhancing systematic innovation and long-term problem-solving. *International Journal of Systematic Innovation*, 9(3).
[https://doi.org/10.6977/IJoSI.202506_9\(3\).0002](https://doi.org/10.6977/IJoSI.202506_9(3).0002)
- Amazon Web Services. (2022). *Amazon EC2 instance network bandwidth*. Amazon Web Services. Accessed March 16, 2026. <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-network-bandwidth.html>
- Babu, C. S., Sekhar, M. R., Sachin, A., & Brindha, B. (2025). Leveraging artificial intelligence techniques in design thinking tools for software engineering. In *Innovative design thinking approaches in software engineering* (pp. 27–52). IGI Global Scientific Publishing.
<https://doi.org/10.4018/979-8-3693-9531-8.ch002>
- Brown, T. (2008). Design thinking. *Harvard Business Review*, 86(6), 84–92. Accessed March 16, 2026. <https://hbr.org/2008/06/design-thinking>
- Chen, L., Song, Y., Ding, S., Sun, L., Childs, P., & Zuo, H. (2024). TRIZ-GPT: An LLM-augmented method for problem-solving. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* (Vol. 6, p. V006T06A010). American Society of Mechanical Engineers.
<https://doi.org/10.1115/DETC2024-143163>
- Chongwatpol, J. (2024). A technological, data-driven design journey for artificial intelligence (AI) initiatives. *Education and Information Technologies*, 29(12), 15933–15963.
<https://doi.org/10.1007/s10639-024-12459-8>
- Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3), 319–340.
<https://doi.org/10.2307/249008>
- Dennehy, D., Schmarzo, B., & Sidaoui, M. (2022). Organising for AI-powered innovation through design: The case of Hitachi Vantara. *International Journal of Technology Management*, 88(2–4), 312–334.
<https://doi.org/10.1504/IJTM.2022.121507>
- Fischer, F., & Gardoni, M. (2024). Comparing different ChatGPT4.0 generated oriented word suggestions as design fixation interferences for the ideation phase in design thinking. In *IFIP International Conference on Product Lifecycle Management* (pp. 207–219). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-031-93323-3_16
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75–106.
<https://doi.org/10.2307/25148625>
- Indriasari, E., Prabowo, H., Gaol, F. L., & Purwandari, B. (2021). The adoption of design thinking, agile software development and co-creation concepts: A case study of digital banking innovation. In *2021 International Conference on Platform Technology and Service (PlatCon)* (pp. 1–6). IEEE.
<https://doi.org/10.1109/PlatCon53246.2021.9680763>
- Ishfaq, F., Marwat, S. N. K., Khan, W. U., Shahzad, S., Khan, S., & Abbasi, Q. H. (2025). A machine learning based empathy mapping framework for enhancing user experience through app review analysis. *Scientific Reports*, 16(1), Article 30729.
<https://doi.org/10.1038/s41598-025-30729-4>
- Livotov, P., Huber, N., Hentschel, C., Mayer, O., Träger, J., Bohn, G., Höcht, L., Kläb, M., Kuhlmann, L., & Scherb, B. (2026). Comparative study on AI-augmented inventive problem solving with TRIZ, TRIZ-AI hybrid, and autonomous AI: An inline coating measurement case in lithium-ion cell production. In *International TRIZ and Artificial Intelligence Conference* (pp. 35–54). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-032-08847-5_3
- Macedo, I., Meireles, M. A. C., Barreto, H., Steinmacher, I., Maldonado, J. C., Gadelha, B., & Conte, T. (2025). AI in the design thinking toolbox: What to recommend, when,

- and why. In *Simpósio Brasileiro de Engenharia de Software (SBES)* (pp. 755–761). SBC/ACM.
<https://doi.org/10.5753/sbes.2025.11538>
- Mysior, M., Iniotakis, C., & Zyga, Ł. (2025). Large language model-based functional modeling of technical systems. In *International TRIZ and Artificial Intelligence Conference* (pp. 227–239). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-032-08847-5_16
- Nada, M., Ayaou, I., Chibane, H., Liverneaux, P., & Cavallucci, D. (2026). A methodological framework integrating TRIZ and AI to address complex design contradictions: Application to an innovative prosthetic thumb. In *International TRIZ and Artificial Intelligence Conference* (pp. 294–307). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-032-08851-2_19
- Obojski, J. (2026). Breaking industry barriers: Using AI to contextualize TRIZ principles across diverse sectors. In *International TRIZ and Artificial Intelligence Conference* (pp. 323–334). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-032-08851-2_21
- OpenSSL Project. (2023). *openssl-speed command documentation*. Accessed March 16, 2026. <https://www.openssl.org/docs/man3.0/man1/openssl-speed.html>
- Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3), 45–77.
<https://doi.org/10.2753/MIS0742-1222240302>
- Qiao, J., Liu, Y., An, Z., & Lee, C.-H. (2026). Requirement and service quality-driven service design for cross-border e-commerce export service optimization incorporated with TRIZ and generative AI. In *International TRIZ and Artificial Intelligence Conference* (pp. 335–352). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-032-08851-2_22
- Redis Labs. (2023). *Redis benchmark results*. Accessed March 16, 2026. <https://redis.io/docs/management/optimization/benchmarks/>
- Savransky, S. D. (2000). *Engineering of creativity: Introduction to TRIZ methodology of inventive problem solving*. CRC Press.
<https://doi.org/10.1201/9781420038958>
- Shin, W.-S., Choi, Y., & Hyun, J. S. (2024). Examining the structural attributes of TRIZ contradiction matrix using exploratory data analysis. *International Journal of Systematic Innovation*, 8(4), 1–17.
[https://doi.org/10.6977/IJoSI.202412_8\(4\).0001](https://doi.org/10.6977/IJoSI.202412_8(4).0001)
- Shneiderman, B. (2020). Human-centered artificial intelligence: Three fresh ideas. *AIS Transactions on Human-Computer Interaction*, 12(3), 109–124.
<https://doi.org/10.17705/1thci.00131>
- Sun, W., Zhao, Y., Xi, C., & Lyu, L. (2026). Technical contradiction extraction based on two-stage knowledge fusion mechanism. In *International TRIZ and Artificial Intelligence Conference* (pp. 308–322). Springer Nature Switzerland.
https://doi.org/10.1007/978-3-032-08851-2_20
- Venable, J., Pries-Heje, J., & Baskerville, R. (2016). FEDS: A framework for evaluation in design science research. *European Journal of Information Systems*, 25(1), 77–89.
<https://doi.org/10.1057/ejis.2014.36>
- Yin, R. K. (2018). *Case study research and applications: Design and methods* (6th ed.). Sage Publications, Thousand Oaks, CA.

Appendix

Appendix A. Representative LLM prompt templates

All five TriGenDT stages. Structure per template: (i) system role → (ii) context (filled with MedNet CDSS scenario content) → (iii) task instruction → (iv) output format → (v) human validation gate.

Model configuration (all stages): instruction-tuned GPT-4-class model; temperature = 0.4; max_tokens = 1,500; top_p = 1.0. Temperature set below 0.7 to favor structured, traceable outputs over maximal ideation diversity.

A.1 Stage 1: Empathize and define (function analysis + ideal final result)

- (i) System role: You are a TRIZ-knowledgeable systems analyst specializing in healthcare information systems. Your task is to extract and cluster user pain points from the documentation provided, identify the functional components of the current system, and draft an IFR statement. You augment human analysis, you do not replace clinical or governance judgment.
- (ii) Context: System: MedNet CDSS, a cloud-native platform across six regional hospital sites delivering real-time ICU sepsis and deterioration alerts, subject to GDPR and HIPAA compliance. Documentation provided: support tickets (alert-delay complaints from ICU nursing staff; consent re-prompting disruptions; data inconsistency across sites; audit-log storage growth; data gaps during patient transfers); incident reports (delayed alert responses; non-compliant encryption flagged in regulatory audit). Stakeholder roles: Software Architect, Clinical Informaticist, Data Protection Officer, ICU Nurse Informaticist.
- (iii) Task instruction: (i) Extract and cluster pain points by frequency and clinical/operational severity; present the five highest-priority clusters with severity label and one-sentence rationale. (ii) Identify key functional components and harmful or insufficient functions. (iii) Draft one IFR statement in the form: “The [system] performs [function] without [harmful effect], while [constraint] is fully satisfied, with [no negative side-effect].” (iv) Decompose the IFR into a maximum of four evaluable acceptance criteria for Stage 5 validation.
- (iv) Output format: Section A: Pain-point clusters (ranked table: Rank | Cluster | Severity | Rationale); Section B: Function model (bulleted list); Section C: Draft IFR statement; Section D: Acceptance criteria (numbered list, ≤4 items, each with measurable threshold).
- (v) Human validation gate: The practitioner team reviews all four sections, approves or modifies each cluster and acceptance criterion, and signs off the IFR statement before Stage 2 begins. No output proceeds without team validation. Decisions and rationale are logged in the stage decision record.

A.2 Stage 2: Problem framing (contradiction matrix)

- (i) System role: You are a TRIZ-knowledgeable software requirements analyst. Your task is to identify technical and physical contradictions in the requirements specification provided. A technical contradiction exists when improving parameter A worsens parameter B. A physical contradiction exists when the same parameter must simultaneously take two incompatible values. You present candidate contradictions for practitioner review; you do not make architectural decisions.
- (ii) Context: System: MedNet CDSS as above. IFR statement (validated at Stage 1): “The CDSS provides sepsis and deterioration alerts within 200 ms of threshold violation, with zero additional latency due to privacy controls, while all data access events are fully auditable and GDPR/HIPAA compliant, with zero net increment of architectural complexity.” Acceptance criteria: alert latency ≤ 200 ms; no unencrypted PII at rest or in transit; 100% access audit trail (GDPR Art. 30); no net increase in architectural complexity. Key NFRs in conflict: real-time alert delivery ≤ 200 ms; AES-256 encryption mandatory; synchronous per-access consent check required; per-access audit logging required; system must not increase component count beyond baseline.
- (iii) Task instruction: Identify up to five candidate contradiction pairs. For each specify: (i) two conflicting requirements as plain-language statements; (ii) TRIZ contradiction type (Technical/Physical); (iii) closest TRIZ engineering parameters with parameter numbers; (iv) severity estimate (Critical/High/Medium/Low) based on architectural centrality and patient-safety impact; (v) one-sentence severity rationale. Present as a ranked table, most severe first. Flag any candidate that may be a configuration issue rather than a true architectural contradiction.
- (iv) Output format: Ranked table, ID | Contradiction description | Type | TRIZ parameters (#X ↔ #Y) | Severity | Rationale | Configuration-issue flag (Yes/No).

- (v) Human validation gate: The practitioner team validates each candidate: confirms or rejects contradiction type and severity, excludes configuration issues (C4 multi-site synchronization excluded as deployment concern), and approves the final contradiction map. All accept/reject decisions and rationale are logged.

A.3 Stage 3: Ideate (40 inventive principles + effects database)

- (i) System role: You are a TRIZ-knowledgeable software architect. For each TRIZ inventive principle provided, generate a software-engineering-specific interpretation and one concrete architecture concept applicable to the system described. You produce a ranked option set for practitioner review; you do not make a final recommendation.
- (ii) Context: System: MedNet CDSS, cloud-native, microservices architecture, six hospital sites, EHR integration via HL7 FHIR API. Architecture constraints: existing microservices topology must be preserved where possible; AES-256 encryption is non-negotiable; Apache Kafka is already deployed; the alert engine must not hold a direct EHR connection; GDPR data minimization (Art. 5) applies. Validated contradiction map (Stage 2): C1 (Critical), Alert delivery latency (≤ 200 ms) vs AES-256 encryption + synchronous consent-check overhead (400–800 ms), Technical #9 Speed $\leftrightarrow$ #24 Loss of Information; C2 (High), Centralized patient cache vs GDPR data minimization, Physical, #26 Quantity $\leftrightarrow$ #25 Loss of Time; C3 (Medium), Real-time audit logging vs throughput, Technical, #33 Ease of Operation $\leftrightarrow$ #39 Productivity. TRIZ Contradiction Matrix recommendation for C1: Inventive Principles #10 Prior Action, #15 Dynamics, #25 Self-Service, #35 Parameter Change.
- (iii) Task instruction: For each of the four recommended principles (#10, #15, #25, #35) generate: (i) software-domain interpretation (2–3 sentences), grounded in MedNet CDSS architecture; (ii) one concrete architecture concept applying the principle to C1, with C2/C3 co-benefits where applicable; (iii) estimated impact on C1 parameters (High/Medium/Low); (iv) GDPR, HIPAA, or governance implications. Rank the four concepts by estimated C1 impact. Do not select or recommend; the practitioner team decides.
- (iv) Output format: Per principle: Principle # and name $\rightarrow$ Software-domain interpretation $\rightarrow$ Architecture concept (name + 3–5 sentence description) $\rightarrow$ Estimated C1 impact $\rightarrow$ C2/C3 co-benefit (if any) $\rightarrow$ Governance implications.
- (v) Human validation gate: The practitioner team reviews all four concepts, selects and shortlists for Stage 4, documents modifications and rejections with rationale. In the MedNet scenario: Concepts A (P#10, pre-staging enclave) and D (P#35, field-level encryption + async audit) were shortlisted; Concepts B and C were logged as future enhancement candidates due to governance complexity. All decisions logged.

A.4 Stage 4: Prototype (Su-Field analysis + trimming)

- (i) System role: You are a TRIZ-knowledgeable software architect specializing in Su-Field functional modeling and Trimming analysis. Your task is to generate functional architecture variants for the shortlisted solution concepts and identify redundant components for trimming. You produce options for practitioner selection, but you do not finalize the architecture.
- (ii) Context: System: MedNet CDSS as above. Shortlisted concepts: Concept A (P#10), purpose-limited in-memory alert enclave, six alert-critical fields (heart rate, respiratory rate, temperature, blood pressure, SpO₂, NEWS2 score), AES-256 in-use encryption, TTL 45 minutes post-discharge; Concept D (P#35), field-level encryption, alert-critical fields separated from EHR, asynchronous Kafka audit log. Current components eligible for trimming: synchronous consent-check middleware; full-record alert cache. Core components to retain: alert engine, EHR system, Kafka event streaming platform.
- (iii) Task instruction: (i) Combining Concepts A and D, generate three Su-Field functional architecture variants for an SAE pattern. For each: describe functional components and relationships; show how harmful interactions (encryption-latency coupling, consent-check overhead) are neutralized; show how beneficial fields (Kafka, existing microservices) are preserved. (ii) Apply TRIZ Trimming: identify components whose functions can be transferred or eliminated without functional loss. Rank three variants by fit with the existing MedNet microservices topology. Present as options, do not select.
- (iv) Output format: Section A, Three SAE architecture variants (Variant # | Component description | Harmful-interaction neutralization | Beneficial-field preservation | Topology fit High/Medium/Low); Section B, Trimming log (Component | Current function | Disposition Trim/Transfer/Retain | Receiving component if Transfer | Rationale).
- (v) Human validation gate: Practitioner team selects one variant and approves trimming decisions. In the MedNet scenario: Variant 2 selected (best topology fit); synchronous consent-check middleware trimmed (function transferred to patient admission workflow); full-record alert cache trimmed (function replaced by SAE enclave). Net: –2 components, +1

SAE microservice. All decisions logged.

A.5 Stage 5: Test and refine (ideal final result check + contradiction re-scan)

- (i) System role: You are a TRIZ-knowledgeable software architect and systems evaluator. Your tasks are: (i) evaluate the proposed architecture against each IFR acceptance criterion; (ii) perform a contradiction re-scan on the architecture specification to identify residual or newly introduced contradictions. You produce an evaluation report for practitioner review; you do not issue a proceed/iterate decision unilaterally.
- (ii) Context: System: MedNet CDSS as above. SAE architecture (Stage 4): Layer 1, alert enclave microservice, six alert-critical fields, AES-256 in-use encryption, pre-populated at ICU admission, TTL 45 min post-discharge, alert engine queries enclave exclusively, estimated query latency ~18 ms (design-level estimate); Layer 2: full EHR encrypted at rest (AES-256) and in transit (TLS 1.3), no direct alert engine connection; Layer 3: Apache Kafka async audit service, at-least-once delivery guarantee, append-only log, zero latency overhead on alert path. IFR acceptance criteria (Stage 1): (i) alert latency ≤ 200 ms; (ii) no unencrypted PII at rest or in transit; (iii) 100% access audit trail (GDPR Art. 30); (iv) no net increase in architectural complexity.
- (iii) Task instruction: (i) IFR check: for each acceptance criterion, assess whether the SAE architecture satisfies it by design. State: the architectural property that satisfies/fails the criterion; projected status ($\checkmark$ * satisfied by design / * partially satisfied or residual issue / $\times$ not satisfied); empirical validation procedure required to confirm. (ii) Contradiction re-scan: identify (a) residual contradictions from the Stage 2 map that remain unresolved; (b) any new contradictions introduced by the SAE design. For each issue: contradiction description, type, severity, and proposed mitigation.
- (iv) Output format: Section A, IFR evaluation table (Criterion | Target | SAE assessment | Projected status | Validation procedure); Section B, Contradiction re-scan results (Issue ID | Description | Type | Severity | Proposed mitigation); Section C, Iterate/proceed options (practitioner team decides).
- (v) Human validation gate: Practitioner team reviews IFR evaluation and re-scan, makes iterate/proceed decision, logs rationale. In the MedNet scenario: all four IFR criteria assessed $\checkmark$ (satisfied by design, pending empirical confirmation); one residual issue identified (SAE TTL gap 2–5 min during ICU-to-ward transfer, *); safe-escalation default approved as interim workaround; proceed decision logged. Full evaluation recorded as a knowledge-base entry.

Appendix B. Full expert appraisal item-level scoring matrix

[Table A1](#) presents the complete process-coherence scoring matrix for the preliminary expert appraisal reported in Section 4.5.

Table A1. Full item-level process coherence scoring matrix (five panelists × five stages × four rubric dimensions)

Stage/dimension	P1	P2	P3	P4	P5	Mean	Min	Max	Range	SD
S1/D1	5	4	4	5	4	4.40	4	5	1	0.55
S1/D2	4	4	5	4	4	4.20	4	5	1	0.45
S1/D3	5	4	4	4	5	4.40	4	5	1	0.55
S1/D4	4	4	4	5	4	4.20	4	5	1	0.45
S2/D1	4	4	4	5	4	4.20	4	5	1	0.45
S2/D2	4	5	4	4	4	4.20	4	5	1	0.45
S2/D3	4	4	5	4	4	4.20	4	5	1	0.45
S2/D4	5	4	4	4	5	4.40	4	5	1	0.55
S3/D1	4	4	4	4	5	4.20	4	5	1	0.45
S3/D2	4	5	4	4	4	4.20	4	5	1	0.45
S3/D3	5	4	4	5	4	4.40	4	5	1	0.55
S3/D4	4	4	4	4	4	4.00	4	4	0	0.00
S4/D1	4	4	5	4	4	4.20	4	5	1	0.45
S4/D2	4	4	4	5	4	4.20	4	5	1	0.45
S4/D3	4	5	4	4	4	4.20	4	5	1	0.45
S4/D4	5	4	4	4	4	4.20	4	5	1	0.45
S5/D1	4	4	4	4	5	4.20	4	5	1	0.45
S5/D2	4	4	5	4	4	4.20	4	5	1	0.45
S5/D3	4	4	4	5	4	4.20	4	5	1	0.45
S5/D4	4	5	4	4	4	4.20	4	5	1	0.45
Overall	—	—	—	—	—	4.22	4	5	1	0.42

Notes: P1–P5: Anonymized panelist IDs; D1: Stage-input completeness; D2: TRIZ tool application correctness; D3: GenAI interaction protocol compliance; D4: Artifact chain integrity; Scale: 1 (does not pass) to 5 (passes fully); SD: Standard deviation across five panelists; Range = Max – Min; All scores 4–5; No panelist scored any cell below 4; Devil’s-advocate instruction and anonymous individual scoring applied before group comparison to mitigate acquiescence bias. Abbreviations: GenAI: Generative artificial intelligence; TRIZ: Theory of Inventive Problem Solving.