An AI Orchestrator Model for Troubleshooting Skills Training in Network Engineering

Kapil Mehta
Cisco Systems
Bengaluru, KA, INDIA

Prashant Sumanprasad Bhadoria
Cisco Systems
Durham, NC, USA

Jaypal Baviskar
Cisco Systems
Bengaluru, KA, INDIA

Abstract

When a network goes down or resiliencies are at stake, the pressure to restore the network to a business-as-usual state is immense. In that moment, engineers often fall into the trap of "solution-jumping" relying on quick fixes, such as reloads or using Generative AI to provide a quick fix rather than digging for the actual root cause. While these one-shot solutions might bring the network back to its desired state, they often fail to solve the underlying problem, which leads to repeat issues and a steady decline in customer trust.

Our method looks at how we can better prepare network engineers for these high-pressure moments. Instead of relying just on traditional, theory-heavy training or only expert-led trainings, we've blended a framework that focuses on cognitive conditioning. we propose the KTO-AI methodology (Kepner-Tregoe, Topology awareness, and OSI-layer mapping) with the 4E's model (Education, Experience, Exposure, and Environment) in a simulated environment, we're shifting the focus from speed to accuracy.

The core of our approach is a mandatory, rigorous "Assess and Acquire" phase that must happen before any "Act" phase is permitted. We've validated this methodology with experts in the network troubleshooting domain, and the results were clear: when engineers receive real-time customer satisfaction scores (CSAT) and Question Credit feedback for each reasoning step, they stop guessing and start solving.

To scale this cognitive conditioning, we propose an AI Agent Orchestrator as a design model to provide Experience, Exposure, and Environment. The orchestrator is intended to automate repetitive training functions: presenting failure scenarios, tracking Question Credit, recording CSAT trends, enforcing Assess/Acquire phase discipline, and providing structured feedback. This enables organizations to run continuous, personalized troubleshooting skills training at scale. We believe that by providing these safe spaces for practice regularly through AI orchestration, organizations can build the technical and psychological resilience their teams need to handle critical incidents effectively, ultimately leading to a more stable network and superior customer experience.

CCS Concepts

• **Social and professional topics → Computing education**; • **Networks** → *Network management*.

Keywords

AI orchestration, network troubleshooting, engineering education, cognitive conditioning, Kepner-Tregoe, work integrated learning

ACM Reference Format:
Kapil Mehta, Prashant Sumanprasad Bhadoria, and Jaypal Baviskar. 2026. An AI Orchestrator Model for Troubleshooting Skills Training in Network Engineering. In *Proceedings of SIGCOMM Education Workshop 2026: Networking Education for the AI Generation (SIGCOMM Edu. 2026).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/XXXXXXX.XXXXXXX

1 Introduction: The Crisis of "Solution-Jumping"

The most consequential moment in a critical network outage is not the failure itself, but the "blind reaction" of the engineer. In the modern era, this is exacerbated by Generative AI. While GenAI can provide a possible solution in seconds, it can encourage shortcuts, moving from a symptom directly to a fix without understanding the underlying issue. This practice of "solution-jumping" can reduce customer trust when the initial fix fails, leaving the root cause unaddressed and the network vulnerable to repeat failures.

This is a limitation of mindset, not capability. Under operational pressure, engineers exhibit well-documented cognitive biases toward heuristic shortcuts rather than systematic analysis [1]. James Reason's framework of human error

SIGCOMM Edu. 2026, Denver, CO, USA

ACM ISBN 979-8-4007-XXXX-X/2026/08
https://doi.org/10.1145/XXXXXXX.XXXXXXX

[2] categorizes such failures as "slips"—execution errors on otherwise sound plans—that are exacerbated when decision-making authority is decentralized (as with GenAI assistance). The core philosophy of the Kepner-Tregoe (KT) methodology [3] is: "Stop guessing, start knowing." Yet, under the pressure of a "Hot Seat" troubleshooting scenario, engineers often revert to guessing. We ground this concern in both theory and observation. Kahneman's work on cognitive biases [1] and Reason's framework [2] document how engineers under pressure exhibit heuristic shortcuts. Our case study also (Section 6) illustrates one instance where an experienced troubleshooter initially defaulted to blind action under operational pressure. To combat this, troubleshooting education must shift from teaching how to fix incidents to teaching a troubleshooting mindset grounded in evidence.

We propose an AI Orchestrator Model for troubleshooting skills training that enforces a rigorous "Assess and Acquire" phase before any "Act" phase is permitted. This approach aligns with current trends in Work Integrated Learning (WIL) [4], which emphasizes the integration of theoretical knowledge with practical, real-world professional application. By automating repeatable practice interactions rather than replacing human mentors for education (the first E, Section 4), the AI Orchestrator can expand access to cognitive-conditioning drills beyond the practical limits of synchronous expert-led sessions to provide Experience, Exposure, and Environment (the remaining three E's, Section 4).

2 The KTO-AI Framework—Engineering Troubleshooting Discipline

2.1 The Primacy of "Assess and Acquire"

The framework is built on one fundamental discipline: the enforced separation between thinking and doing. This is the essence of the 4A's Loop, adapted from the Kepner-Tregoe methodology [3].

Each phase serves a specific purpose:

Assess (Situation Appraisal): The engineer must establish business impact and Topology Awareness. Before analyzing logs, the engineer must understand the physical and logical path of impacted traffic, the network segment affected, and the business impact of the outage. This context prevents misdirected investigation and ensures all stakeholders understand the scope.

Acquire (Data Gathering): The engineer systematically gathers evidence using two complementary techniques: OSI-Layer Mapping (checking each layer from Physical to Application) and "Is/Is Not" analysis (isolating variables that correlate with the failure). This phase reduces the problem space from infinite possibilities to a finite set of probable causes.

Analyse (Synthesis): Only after acquiring sufficient evidence does the engineer form a hypothesis about the probable root cause. Critically, this hypothesis must be grounded in the acquired data, not in intuition, pattern matching, or AI-generated suggestions.

Act (Verification and Restoration): The engineer executes targeted actions to confirm the hypothesis, restores service, and documents the root cause for network-wide prevention.

The critical discipline is this: No "Act" phase should be permitted until "Acquire" is sufficiently complete. Under operational pressure, this boundary erodes—engineers skip directly to blind fixes to restore service quickly. This enforced separation is what prevents that erosion and preserves the evidence needed for root-cause diagnosis. It is a structural requirement enforced by the AI Agent Orchestrator (discussed in Section 5).

Why this matters: Once an engineer reboots a router to restore service, volatile evidence such as RAM contents, log buffers, and ARP tables can be lost. The organization is left with a temporarily restored service but no understanding of the underlying mechanism. This opens the door for the same failure to recur, and the next troubleshooting session begins from zero knowledge.

The discipline trains engineers to pause and ask the critical question: "Do I possess sufficient evidence to form a defensible hypothesis? Or am I operating on intuition, pattern recognition, or an unvalidated AI suggestion?"

2.2 Restoration vs. Evidence Preservation

A secondary but critical principle is the ability to decide when to prioritize Service Restoration and when to prioritize Log Collection.

The Blind Fix: Rebooting a device to restore service immediately destroys volatile evidence.

The Informed Fix: Capturing the state of the device or network first, then restoring service. This protects the "diagnostic evidence" required for network-wide prevention of repeat incidents.

By practicing this discipline in a safe environment, engineers learn the psychological and technical resilience required to resist the urge to "quick fix" and instead prioritize understanding.

3 The Troubleshooting Framework—Logic Over Action

3.1 The 4A's Methodology

We utilize a streamlined version of the KT process [3], organized into the 4A's:

Assess: Situation Appraisal. Defining the problem clearly and understanding business impact.

Acquire: Data gathering using technical tools to isolate variables.

Analyse: Synthesis. Forming a hypothesis based on acquired data.

Act: Verification and resolution, with documentation for network-wide learning.

3.2 Technical Tools for Isolation—Topology and OSI

To move from guessing to knowing, the engineer must use two primary technical anchors during the "Acquire" phase:

Topology Awareness: The engineer must map the physical and logical path of the traffic. Understanding the topology prevents the engineer from searching for a problem in the wrong segment of the network.

OSI-Layer Mapping: By systematically checking which OSI layer the problem lies in (starting from Physical Layer 1 up to Application Layer 7), the engineer ensures that no fundamental failure is overlooked.

3.3 Question Credit and Customer Trust

Troubleshooting is not just a technical task; it is a customer experience task as well. We use the concept of Question Credit, a finite budget of customer patience in our training model.

Trust Gains: Every high-value, logical question in the Assess and Acquire phases that narrows the problem space increases the customer's confidence in the engineer's competence.

Trust Losses: Every unrelated question or "blind action" (e.g., "Let's try rebooting the router") that fails to fix the problem reduces customer trust irreversibly.

When an engineer "solution-jumps," they spend their Question Credit too early. If the fix fails, the customer loses confidence, regardless of certifications or prior success. This metric is critical for training because it teaches engineers that patience and evidence are more valuable than speed.

The combination of framework (4A's, Section 3.1), metrics (CSAT and Question Credit), and pressure (time constraints, stakeholder anxiety) creates cognitive conditioning. The metrics provide real-time feedback that mirrors actual customer interactions, while the pressure tests whether engineers can maintain systematic discipline under stress. This integration—structure plus feedback plus pressure is what enables learning, not the framework alone.

4 The 4E's Model—Cognitive Conditioning Through Simulated Pressure

To prepare an engineer for a high-stakes outage, theoretical knowledge alone provides limited skill development. Contemporary research in learning design [4] emphasizes the integration of excellence and innovation across multiple dimensions of the learning experience. Drawing on this framework and aligned with Work Integrated Learning principles [4], we implement the 4E's Model to build professional resilience across two delivery modes: Human mentors can provide Education (KT theory, frameworks, psychological concepts) and guide interpersonal coaching. The AI Orchestrator provides enviornment and automates Experience and Exposure the repetitive failure scenarios and time-pressured drills.

Education: Instruction in KT methodology [3], Topology awareness, OSI mapping, and the psychological aspects of incident response. This provides the theoretical foundation by human mentors or other available resources papers, videos etc.

Experience: Practical application of the 4A's in controlled failure scenarios that mirror production complexity. Engineers apply knowledge to realistic problems.

Exposure: Simulating the psychological pressure of a critical incident (time constraints, stakeholder pressure, declining CSAT) to test the engineer's ability to remain systematic under stress. This builds psychological resilience.

Environment: A safe, simulated space where "failing" (e.g., spending all Question Credit on blind guesses) is used as a learning tool to demonstrate the real-world consequences of solution-jumping. This environment allows experimentation without production risk.

Our initial validation used a human mentor as the Orchestrator, implementing this environment by blocking students who attempted to skip the "Assess and Acquire" phases and providing immediate feedback, CSAT updates, and Question Credit visibility. The results validated the model's effectiveness. Grounded in Kolb's experiential learning theory [5], which posits that learning occurs through concrete experience, reflective observation, abstract conceptualization, and active experimentation, and reinforced by Ericsson's research on deliberate practice [6], which emphasizes real-time feedback as critical for skill acquisition in complex domains,

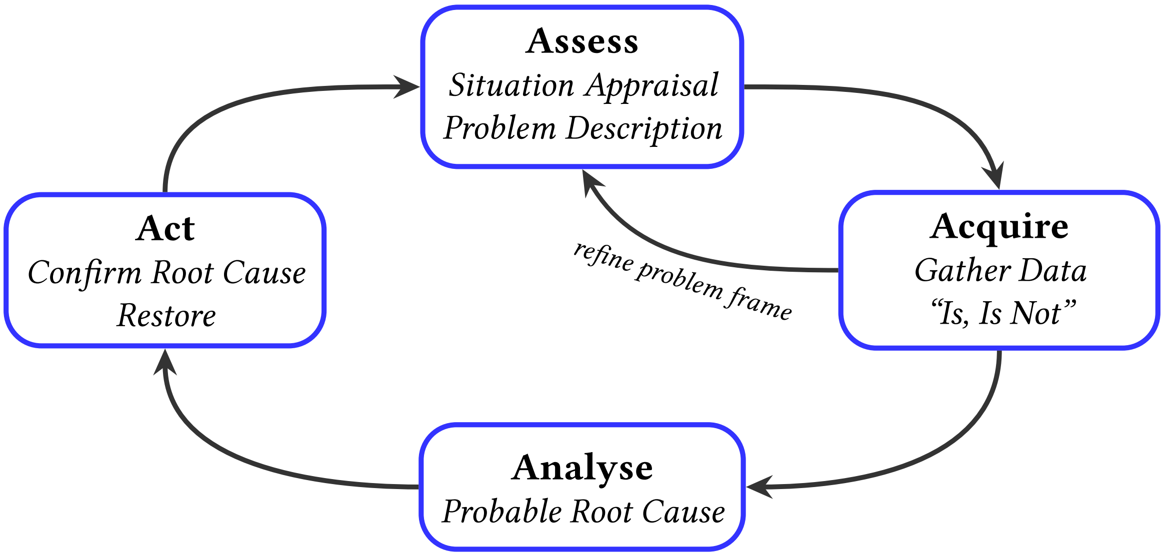

Figure 1: The 4A's summarized Kepner-Tregoe troubleshooting loop. The additional Acquire-to-Assess loop emphasizes iterative refinement before proceeding to Analyse and Act.

we demonstrate that engineers can develop cognitive resilience through structured conditioning. However, human mentors do not scale. This is why we propose the AI Agent Orchestrator.

5 The AI Agent Orchestrator—Automating Cognitive Coaching at Scale

5.1 The Problem—Human Mentorship Does Not Scale

A single expert mentor can effectively train a limited number of engineers using 1:1 or small group sessions. For organizations with hundreds or thousands of engineers, this approach is impractical in the age of AI. Furthermore, inconsistency in feedback quality across multiple mentors leads to variable learning outcomes.

5.2 The Solution—Automating training tasks to provide Experience, Exposure and Enviornment

We propose an AI Agent Orchestrator that automates the mentorship experience. Critically, the AI Agent is NOT a "GenAI that provides answers." Instead, it is a "Cognitive Coach" that:

- **Provides limited problem statements:** The AI agent mimics production-grade networking problem statements by providing very limited information initially, encouraging students to ask the right questions for Assess and Acquire.
- **Provides Real-Time CSAT Scoring:** At each question or action the student proposes, the AI evaluates whether it is evidence-based or a "guess," and provides immediate feedback (0–10 scale).
- **Tracks Question Credit Budget:** The AI maintains a visible Question Credit meter, showing the student how many high-quality inquiries remain before the customer (simulated or real) loses patience.
- **Delivers Tailored Guidance:** When a student asks for help or makes a poor decision, the AI provides feedback ("Why do you think reloading the router will solve this? What evidence supports that?") rather than direct answers.
- **Generates Personalized Scenarios:** The AI creates diverse failure scenarios (configuration drifts, capacity exhaustion, protocol bugs).
- **Documents Learning Outcomes:** The AI generates a post-incident report for each simulation, including the student's reasoning quality, time to diagnosis, and recommendations for improvement.

5.3 Scaling from 1:10 to 1:1000+

By integrating AI agents with expert guidance, the AI Agent Orchestrator enables:

- **Continuous Availability:** Engineers can access modules, at their own pace.
- **Consistent Quality:** All engineers receive identical feedback standards, eliminating variability.

- **Personalization at Scale:** The AI learns from each engineer's performance and tailors scenarios to their weak points.
- **Organizational Scaling:** Easy to scale and compare with human trainer baselines.

6 Case Study of Human Orchestration

A human-orchestrated exercise was conducted with 6 experienced network troubleshooters to prevent solution-jumping and enforce disciplined diagnostic reasoning under real production-grade pressure. This section presents the complete case study, demonstrating how cognitive conditioning through structured inquiry transforms reactive guessing into systematic problem-solving.

6.1 Scenario and Problem Statement

A user reported critical degradation of a real-time video conferencing application (VCA) used for business meetings conducted from home. The initial problem statement was vague and symptom-focused:

> "VCA meetings at home degrade severely. During these times, I can see VCA trying to reconnect, my audio cuts out, video cuts out, and becomes unusable. This is having a serious impact to my business as I cannot have proper customer meetings."

The troubleshooting exercise began with this problem statement and no other context. The participant's challenge: diagnose the root cause while maintaining customer trust.

6.2 The Questioning Progression—Tracking CSAT and Question Credit

The framework enforces a rigorous sequence of questioning phases: Assess → Acquire → Analyse → Act. Each question was tracked for its impact on customer satisfaction (CSAT) and remaining "Question Credit," a finite measure of customer patience.

6.3 The Problem Definition Emerges Progressively

As each question was answered, the problem statement evolved, moving from vague symptom to precise, bounded mechanism:

This progressive refinement demonstrates a core principle of the KTO-AI framework: the problem definition becomes precise only through structured evidence gathering [3]. Guessing at the root cause without this progression would have led to the wrong diagnosis.

6.4 The Critical Moment—When the Framework Prevented Solution-Jumping

After Q4, the troubleshooter had identified that multiple applications (VCA, VPN, web browser) were failing simultaneously. Under operational pressure, with declining CSAT (2/10), the natural instinct was to attempt a blind action: "Reload the router and GPON."

Why This Action Almost Happened:

- Multi-app failures appeared to indicate a network infrastructure problem
- Pattern matching (common pattern: multiple apps fail → router issue) overrode incomplete evidence
- Pressure to restore service quickly felt safer than asking more questions

What Actually Happened:

The framework's structural enforcement of the Assess → Acquire → Analyse → Act sequence prevented this action. Instead, Q5 was asked: "Does this happen on a mobile phone?"

The Answer Changed Everything:

"No—when using a mobile phone on the same home network, VCA works perfectly."

The Impact:

- One question eliminated 80% of possible causes (ISP, router, access point, regional service outages)
- CSAT recovered from 2/10 → 4/10 (customer saw logical thinking, not guessing)
- Investigation pivoted from infrastructure to device-specific context
- If the router reboot had been executed, logs would have been destroyed without any improvement

6.5 Evidence Anchoring—The Is/Is Not Matrix

Through questions Q5–Q12, the troubleshooter populated a structured "Is/Is Not" matrix [3], the core tool for evidence-based reasoning. This matrix prevented assumption-based diagnosis:

This matrix made one fact undeniable: the problem occurred ONLY when all five conditions were simultaneously present. No single-cause hypothesis (VCA bug, ISP issue, Wi-Fi interference, VPN config) could explain this multi-variable correlation.

6.6 Root Cause Analysis—Layer 1 vs. Layer 7

The final questions revealed a critical insight: the observed symptoms and the actual fault were at different OSI layers.

Why This Matters:

Table 1: Progression of Questions and Feedback

Turn	Question / Action	CSAT	Q's Left	Key Note
Q1	"Vendor(a) laptop or Windows PC?"	3	10	Device identified
Q2	"Only VCA or other platforms too?"	3	8	Multi-app: affects other VCA app too
Q3	"When did it start?"	2	6	3 months ago, customer impatient
Q4	"VPN/browser issues when VCA fails?"	2	6	VPN also cuts out
Q5	"Does it happen on a mobile phone?"	4	6	NOT on other devices; HIGH-VALUE Q
ACTION	"Reload router and GPON"	2	5	Blind action; cost 1 credit
Q6	"Only at home? Ping google.com?"	1	4	CSAT crashes to 1
Q7	"Successful calls on other devices, same internet?"	3	4	Empathy; credit RETAINED
Q8	"Built-in or external mic/camera?"	4	4	Topology probing
Q9	"What changed 3 months ago?"	5	3	USB-C cable replacement found
Q10	"VPN with or without?"	5	2	Clarification; cost 1 credit
Q11	"Vendor(a) laptop standalone without external monitor?"	7	2	BREAKTHROUGH; problem isolated
Q12	"Different USB-C cable or port?"	8	2	Different cable fixes it
Q13	"Vendor(a) laptop fan speed high? Heat causing issue?"	7	1	Unsupported hypothesis
Analysis	"Strong KT correlation; USB-C cable is root cause"	9	0	All credit spent
Q14	"Cable broken or wire damage?"	9	0	Final verification

Table 2: Problem Definition Evolution

Stage	After Questions	Problem Definition Refined To
1	Q1–Q2	VCA meetings degrade on Vendor(a) laptop; affects other real-time communication tools
2	Q3–Q4	Problem began 3 months ago; VPN and web browsing also affected
3	Q5	Problem is NOT on a mobile phone on same home network—specific to Vendor(a) laptop
4	Q6–Q7	Problem at home only; other devices can use same internet successfully
5	Q8–Q9	External monitor and camera via USB-C; cable replaced 3 months ago
6	Q10	Problem disappears when Vendor(a) laptop is standalone; isolated to external connection
7	Q11	Problem persists with different USB-C port but different cable fixes it
8	Q12	Cable specification: USB-C 3.2; mechanism research begins

Table 3: Is/Is Not Matrix—Final State

Variable	IS (Observed)	IS NOT (Not Observed)
Device Affected	Vendor(a) laptop	a mobile phone, other devices on same network
Applications Affected	VCA, real-time tools, VPN, web browser	Email client, file transfer apps
Network Conditions	2.4 GHz Wi-Fi; mobile hotspot	5 GHz Wi-Fi; wired Ethernet
External Hardware	Monitor + USB-C 3.2 docking station	Vendor(a) laptop standalone; monitor disconnected
Specific Cable	Right-angle USB-C 3.2 cable (original)	Different USB-C cable; different port

Without structured OSI-layer discipline, the troubleshooter's intuition would have pointed to Layer 7 or Layer 3 (application or network layer issues). The actual fault—a physical-layer RF coexistence problem—is counterintuitive and would never have been discovered through pattern matching alone.

Table 4: Symptom vs. Fault—OSI Layer Mapping

Aspect	OSI Layer	Details
What the User Saw (Symptom)	Layer 7 (Application)	VCA reconnects; audio/video cut out; web browser stalls
Where the Problem Originated (Fault)	Layer 1 (Physical)	EMI from USB 3.2 cable; right-angle connector shielding failure; RF leakage into 2.4 GHz Wi-Fi

Table 5: Framework Impact on Troubleshooting Behavior

Aspect	Without Framework	With KTO-AI Framework
First Instinct	Reload router (blind guess after Q4)	Continue systematic inquiry (enforce Q5 first)
CSAT Outcome	Stalled at 2/10 (action failed)	Recovered to 9/10 (logical inquiry succeeded)
Root Cause Identified	No	Yes (USB-C EMI at physical layer)
Evidence State	Destroyed (logs/ARP wiped)	Preserved (matrix documented)

Verification and Resolution:

The troubleshooter replaced the USB-C cable with a certified, shielded straight-connector cable rated for EMI suppression. Packet loss dropped to 0% immediately. Degradation did not recur across 72 hours of continuous VCA usage, web browsing, and VPN connectivity.

6.7 Learning Outcomes—What the Framework Teaches

The Behavioral Shift:

The participant transformed from reactive pattern-matching ("Multiple apps failing = router problem") to systematic evidence-based reasoning ("Device-specific failure = local environment problem"). This shift is not intuitive; it requires cognitive conditioning through the 4E's Model [4]: Education (KT theory [3]), Experience (applying 4A's), Exposure (simulated pressure), and Environment (safe space to learn).

The Value of the Framework:

By enforcing structured phases and high-value questioning, the framework:

- Maintained customer trust even during investigation
- Preserved diagnostic evidence
- Enabled root-cause identification
- Created organizational knowledge (USB-C cable guidelines) for preventing similar incidents

7 Implications for the AI Orchestrator

The case study demonstrates that the human orchestration model works; enforcing the 4A's loop, adapted from KT discipline [3], produces better outcomes than solution-jumping. The AI Agent Orchestrator simply automates this proven discipline, making it accessible to every engineer, every day. This section outlines the intended architecture, actual implementation and validation remain future work.

In the AI-orchestrated version of this scenario:

- The AI would present the scenario in a simulated environment with time pressure (e.g., "Service is degraded. Your customer is on a call with your manager. You have 15 minutes to propose a fix.").
- Each question or action the engineer proposes would generate a real-time CSAT score and Question Credit deduction/addition.
- The AI would block any "Act" phase (e.g., "Reload the router") unless the "Acquire" phase was sufficiently complete, or provide justification if not taking the Act step.
- Upon completion, the AI would generate a post-incident report showing the engineer's reasoning quality, areas for improvement, and similar scenarios to practice.
- The engineer could repeat this scenario as many times as desired, with the AI tracking improvement over time.

8 SCOPE AND FUTURE DIRECTIONS

This research establishes foundational insights through human-led orchestration, with the AI Orchestrator framework ready for implementation. Our initial findings on engineer problem-solving patterns emerge from a focused case study, expanding to a broader population survey will strengthen the generalizability of these insights. Language model integration introduces important considerations, including potential hallucination in performance metrics and grading accuracy, which our proposed validation framework is designed to address. Subsequent phases will systematize these safeguards through larger-scale population studies, controlled experiments, and full orchestrator deployment, creating a robust evidence base for the model's effectiveness in real-world training environments.

9 Conclusion

The convergence of high-pressure operational demands and the accessibility of GenAI has created a critical educational challenge: how do we prepare engineers to resist the temptation to "solution-jump"? Our answer is cognitive conditioning through the 4E's Model [4], enforced by a disciplined framework that prioritizes diagnostic rigor over restoration speed.

By combining the structured rigor of the Kepner-Tregoe methodology [3] with practical exposure to high-pressure scenarios, we have demonstrated (through human orchestration) that engineers can develop the discipline to ask the right questions, preserve diagnostic evidence, and achieve mechanistic understanding.

To scale this proven cognitive conditioning beyond the limitation of human mentorship, we propose an AI Agent Orchestrator that scale and automates the mentor's role. The AI Orchestrator provides real-time CSAT feedback, tracks Question Credit budgets, enforces phase-gate discipline, and delivers personalized scenarios, enabling organizations to run continuous, safe troubleshooting skills training at scale (1:1000+).

We believe that by providing these safe spaces for practice regularly through AI orchestration, organizations can build the technical and psychological resilience their teams need to handle critical incidents effectively, ultimately leading to more stable networks and superior customer experience.

10 Acknowledgement

We sincerely thank Peter T. K. Lee (Leader, Customer Delivery, Cisco Systems) for leading and serving the human orchestrator in this study. We also extend our gratitude to the network troubleshooting practitioners who participated in the sessions, especially those who contributed to the VCA case study.

Finally, we thank the ACM SIGCOMM community for the opportunity to present this work at the SIGCOMM Education Workshop 2026, Networking Education for the AI Generation, where pedagogical innovation in network engineering is increasingly important.

References

[1] Daniel Kahneman. *Thinking, Fast and Slow.* Farrar, Straus and Giroux, New York, NY, USA, 2011.

[2] James Reason. *Human Error.* Cambridge University Press, Cambridge, UK, 1990.

[3] Charles H. Kepner and Benjamin B. Tregoe. *The New Rational Manager: An Updated Edition for a New World.* Princeton Research Press, Princeton, NJ, USA, 1981.

[4] S. Aggarwal, J. Baviskar, and A. S. Mulla. Excellence and innovation in learning and design. In K. Bindumadhavan and N. Lacey, editors, *Work Integrated Learning, Directions for the Future*, volume 1206 of *Lecture Notes in Networks and Systems.* Springer, Singapore, 2025.

[5] David A. Kolb. *Experiential Learning: Experience as the Source of Learning and Development.* Prentice-Hall, Englewood Cliffs, NJ, USA, 1984.

[6] K. Anders Ericsson. Deliberate practice and the acquisition and maintenance of expert performance in medicine and related domains. *Academic Medicine*, 82(10):S1–S2, October 2007.