Human Factors in Artificial Intelligence and Social Computing, Vol. 203, 2026, 30–40
https://doi.org/10.54941/ahfe1007315

Governing the Transition to Action: An Agentic Architecture for Situation-Aware LLMs

Michael Hildebrandt

Institute for Energy Technology, Halden, Norway

ABSTRACT

This paper proposes a framework of ten architectural requirements for AI decision support in safety-critical domains. Derived from Endsley's SA model and the distributed SA perspective, the requirements span real-time perception, structured comprehension, projection, temporal depth, transparency, operator state modelling, auditability, governed activation, inter-agent coherence, and self-monitoring. We evaluate three classes of LLM-based architecture against these requirements and demonstrate that only agentic workflows with knowledge graph grounding can satisfy them. A technical architecture for a simulated air traffic control environment demonstrates how each requirement maps to concrete infrastructure components, and identifies which requirements are readily met and which remain open research challenges.

Keywords: Situation awareness, Artificial situation awareness, Agentic AI, Knowledge graphs, Human-AI teaming, Air traffic control, Large language models

INTRODUCTION

Air traffic control, nuclear process control, and emergency medicine all require operators to integrate information from heterogeneous sources under time pressure. Situation awareness (SA) is central to performance in such domains. Endsley (1995) defined SA as the perception of relevant elements in the environment, the comprehension of their meaning, and the projection of their future status.

Large language models (LLMs) are increasingly investigated as decision support tools for safety-critical operations. Applications under investigation include automated analysis of incident reports (Chen et al., 2024), generation of training scenarios (Gould et al., 2025), and retrieval of procedural guidance (Zhang, Yang, and Zeng, 2024). However, much of this work builds on relatively simple LLM integration patterns, typically a foundation model responding to user queries, or a retrieval-augmented generation (RAG) pipeline that supplies the model with relevant document fragments. These patterns were developed primarily for information retrieval and content generation, not for real-time operational support where errors carry severe consequences.

Received February 6, 2026; Revised April 2, 2026; Accepted April 21, 2026; Available online July 20, 2026

AI tools may degrade rather than support operator SA if poorly integrated into the operational context (Endsley, 2023; Endsley, 2017). Effective human-AI teaming requires taskwork SA (understanding the domain), agent SA (understanding the AI's reasoning), and teamwork SA (understanding coordination between human and AI). These demands extend beyond interface design to the underlying architecture.

We argue that supporting operator SA requires the AI system itself to maintain something functionally analogous to situation awareness. We term this set of requirements artificial situation awareness (aSA) and contend that current LLM integration patterns cannot satisfy them. Instead, agentic workflows with knowledge graph grounding are needed. The primary contribution of this paper is conceptual and architectural. We derive ten aSA requirements from established SA theory, evaluate three classes of LLM architecture against them, and present a technical architecture in a simulated air traffic control environment that demonstrates how the requirements translate to infrastructure components.

THEORETICAL FOUNDATIONS

Situation Awareness in Human-AI Teams. Endsley (2023) extended the three-level SA model to human-AI teams, identifying taskwork SA (the operational domain), agent SA (the AI system's behaviour and reliability), and teamwork SA (coordination between human and AI). That work distinguishes transparency (real-time availability of system state) from explainability (retrospective model-building), both of which impose different architectural requirements.

The distributed SA perspective (Salmon et al., 2009; Stanton et al., 2006) argues that SA in complex sociotechnical systems is distributed across human and non-human agents. The SA-based agent transparency (SAT) model (Chen et al., 2018) complements this perspective, proposing that autonomous systems should communicate their perception (SAT Level 1), reasoning (SAT Level 2), and projections and planned actions (SAT Level 3).

Drawing on these frameworks, we propose that an AI system intended for safety-critical decision support should satisfy a set of functional requirements that we term artificial situation awareness (aSA). The term does not imply awareness in a phenomenological sense. Instead aSA is a set of engineering requirements that can be evaluated against a concrete architecture. The requirements were derived through systematic mapping of established SA theories to the architectural capabilities needed for computational SA. Endsley's (1995) three-level model provided the foundational structure, yielding aSA-1 through aSA-3. The distributed SA perspective (Salmon et al., 2009), the SAT model (Chen et al., 2018), and the levels of automation framework (Parasuraman et al., 2000) contributed requirements addressing the additional demands of human-AI teaming in safety-critical operations. Requirements aSA-4 and aSA-7 address persistence and traceability needs that arise when SA must be maintained computationally rather than cognitively. We do not claim exhaustiveness,

but the set covers the principal architectural concerns identified across these traditions. The requirements are:

aSA-1: **Real-time perception**. The system must continuously ingest and process multi-modal data streams from the operational environment, maintaining a current representation of relevant state variables.

aSA-2: **Structured comprehension**. The system must integrate perceived data into a coherent operational picture using domain knowledge, including anomaly detection, pattern recognition, and context-dependent interpretation.

aSA-3: **Projection**. The system must model future states based on its current understanding, including identification of emerging risks and estimation of time horizons for significant events.

aSA-4: **Temporal depth**. The system must maintain access to operational history at multiple timescales, from seconds to weeks.

aSA-5: **Transparency**. The system's internal state must be available for inspection by human operators in a form appropriate to the operational context (Endsley, 2023; Chen et al., 2018).

aSA-6: **Operator state modelling**. The system must maintain an ongoing estimate of each human operator's cognitive state, including workload, attentional focus, current SA level, and trust calibration. This model informs transparency delivery (aSA-5: what to present, when, in what form) and activation governance (aSA-8: whether the operator can absorb a recommendation or needs more directive support).

aSA-7: **Auditability**. The system must maintain complete provenance chains from data sources through reasoning steps to outputs, including the state of the knowledge base at the time of each decision, supporting both real-time monitoring and post-hoc analysis.

aSA-8: **Governed activation**. The system must implement explicit, inspectable rules governing the transition from situation assessment to action initiation. For any identified situation, the system must determine which class of response is appropriate (from passive informing through recommendation to autonomous action) based on assessed severity, time pressure, consequence reversibility, the current allocation of authority between human and machine, and the system's own confidence assessment (aSA-10). The specific response classes and the upper bound of system autonomy are domain-dependent. The activation boundary must itself be transparent and auditable.

aSA-9: **Inter-agent coherence**. In a multi-agent system, individual agents develop partial, potentially conflicting situation assessments. The system must maintain shared representations across agents, detect inter-agent conflicts, and converge on a coherent joint operational picture. This differs from self-monitoring (aSA-10) because self-monitoring concerns the quality of individual processing, while inter-agent coherence concerns collective consistency. In cross-domain deployments where agents are grounded in different domain ontologies (e.g. ATC separation standards vs. airline fuel policies), agents must reconcile assessments at shared decision points without requiring a single unified world model.

aSA-10: **Self-monitoring**. The system must continuously assess the quality and reliability of its own processing. This includes confidence calibration of its assessments, detection of degraded inputs or processing components,

recognition of situations at or beyond competence boundaries, and consistency checking across its own processing agents. When self-assessment indicates degraded or uncertain SA, this must propagate to both the transparency layer (aSA-5, so the human operator is informed) and the activation governor (aSA-8, so response behaviour shifts toward conservatism).

Research on SA-related accidents consistently finds that operators involved in failures often did not recognise their own SA degradation (Jones and Endsley, 1996). For an artificial system, we have the architectural opportunity to engineer this function explicitly.

The requirements describe a system-level property, not the capability of a single model. The requirements are interdependent. Auditability (aSA-7) requires persistent state (aSA-4), operator modelling (aSA-6) informs transparency (aSA-5) and activation governance (aSA-8), inter-agent coherence (aSA-9) is a precondition for self-monitoring (aSA-10), which in turn is a precondition for both aSA-5 and aSA-8. Mapped to Endsley's (2023) SA-HAT categories, aSA-1 through aSA-4 support taskwork SA, aSA-5 and aSA-7 support agent SA, and aSA-6 and aSA-9 support teamwork SA.

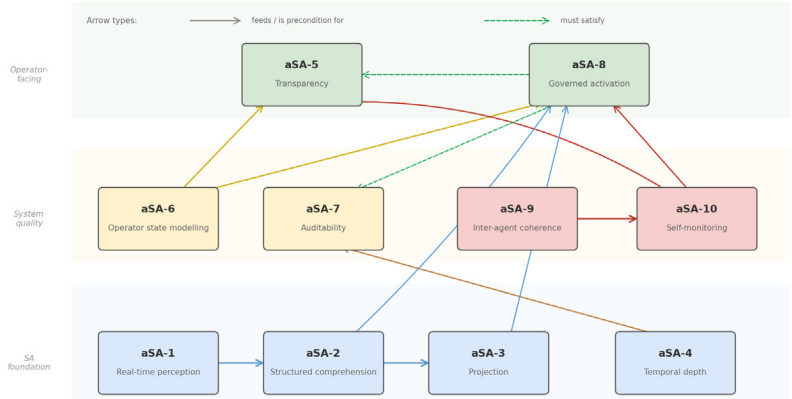

Figure 1: Dependency structure of the ten aSA requirements, organised by functional tier. Solid arrows indicate "feeds / is precondition for", dashed arrows indicate "must satisfy."

EVALUATION OF CURRENT LLM APPROACHES

Foundation Models. A foundation LLM used as a conversational interface operates without persistent connection to the operational environment, failing aSA-1. It has no persistent memory across interactions (failing aSA-4), and its implicit knowledge is neither structured nor inspectable (failing aSA-5 and aSA-7). Hildebrandt (2026) characterised this as expertise without experience, in that the model possesses domain knowledge but lacks grounding in the operational situation where that knowledge must be applied. A foundation model cannot track operator workload (aSA-6), govern its own activation (aSA-8), or as a monolithic system support inter-agent coherence (aSA-9) or self-monitoring (aSA-10).

Retrieval-Augmented Generation. RAG systems (Lewis et al., 2020) address some limitations by retrieving document fragments from an external knowledge base, reducing hallucination and allowing updates without retraining. However, RAG was designed for information retrieval, not operational awareness. Retrieval based on semantic similarity may miss operationally critical information, and context is treated as static documents rather than live state. Graph-based RAG (Edge et al., 2024; Peng et al., 2024) improves knowledge structure but does not address real-time perception, projection, or persistent state. RAG systems lack governed activation, operator modelling, inter-agent coherence, and self-monitoring.

Agentic Workflows with Knowledge Graphs. Multi-agent architectures with knowledge graph grounding can in principle satisfy all ten requirements. Dedicated perception agents address aSA-1, graph-structured layers support aSA-2 through aSA-4, workflow graphs and execution logs provide aSA-5 and aSA-7 as structural properties. Persistent user interaction state enables operator modelling (aSA-6), explicit authority rules enable governed activation (aSA-8), structured message passing with a supervisor agent supports inter-agent coherence (aSA-9), and multi-agent cross-checking provides a structural basis for self-monitoring (aSA-10). Table 1 summarises the assessment.

Table 1: Assessment of LLM architecture classes against aSA requirements (aSA-1 through aSA-10). Ratings indicate in-principle capability.

aSA Requirement	Foundation LLM	RAG	Agentic + KG
aSA-1: Real-time perception	Not supported	Not supported	Supported
aSA-2: Structured comprehension	Minimal	Partial	Supported
aSA-3: Projection	Minimal	Not supported	Supported
aSA-4: Temporal depth	Not supported	Partial	Supported
aSA-5: Transparency	Not supported	Partial	Supported
aSA-6: Operator state modelling	Not supported	Not supported	Supported
aSA-7: Auditability	Not supported	Partial	Supported
aSA-8: Governed activation	Not supported	Not supported	Supported
aSA-9: Inter-agent coherence	Not supported	Not supported	Supported
aSA-10: Self-monitoring	Not supported	Not supported	Supported

TECHNICAL ARCHITECTURE: AGENTIC INFRASTRUCTURE FOR SIMULATED ATC

This section presents a technical architecture that demonstrates how aSA requirements translate into concrete infrastructure. The architecture is situated in a multi-user ATC simulation environment, consisting of multiple controller workstations managing sectors, coupled to flight simulator clients generating realistic traffic and pilot behaviour.

The architecture uses specific tools (n8n for workflow orchestration, Neo4j for graph storage) to make the design concrete, though equivalent infrastructure could be substituted. A weather deviation scenario serves as a running example through the subsections.

Orchestration Layer. The system uses an event-driven workflow orchestrator as the coordination backbone. Simulation events such as trajectory updates, new METARs, pilot voice transmissions arrive as webhook triggers that initiate processing workflows. Each agent is a node in the workflow graph: An LLM call with structured prompt and tool access, a deterministic function (e.g., geometric separation calculation), or a hybrid. The orchestrator manages agent invocation sequences, branching logic, and execution state.

Two properties of this design directly satisfy aSA requirements. First, the workflow graph is itself a transparency artefact (aSA-5). The orchestrator's visual canvas shows exactly what processing occurred, in what order, with what inputs and outputs. The execution log produced by every workflow run satisfies auditability (aSA-7) as a structural property of the orchestration pattern.

Example: the surveillance monitor detects a trajectory change from Aircraft A (deviation left of course), triggering the conflict assessment workflow. The orchestrator invokes weather processing, intent recognition, conflict detection, and projection agents in sequence, with explicit data passing between steps.

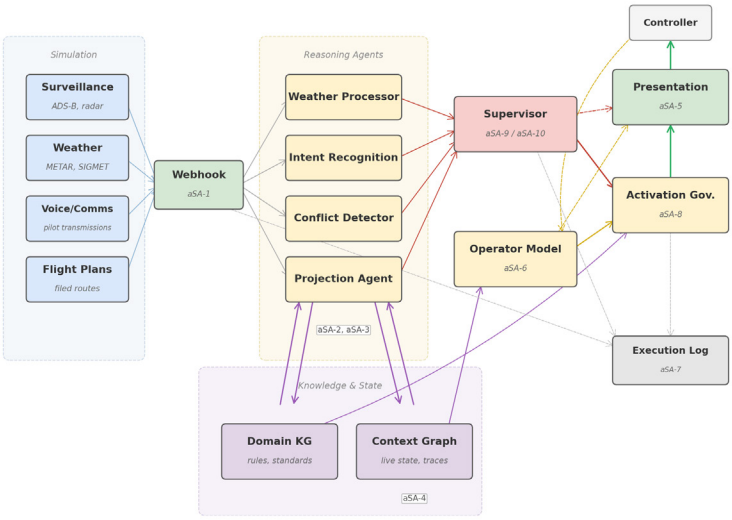

Figure 2: Architecture of the agentic ATC decision support system. Data flows left to right from simulation sources through reasoning agents, supervisor, and activation governor to the presentation layer. aSA requirement mappings are annotated on each component.

Knowledge and State Management. The system maintains two complementary graph layers in a graph database. The domain knowledge graph encodes operational rules (separation standards, airspace classifications, performance envelopes, procedure libraries, and authority rules) with write access restricted to expert review. The context graph maintains live operational state, such as current traffic picture, active conflicts, resolution history, and decision traces. Perception agents write to the context graph as events unfold, and every mutation carries a timestamp and the writing agent's identifier, creating an automatic provenance chain (aSA-7).

This separation enables comprehension and projection agents to query both layers: the domain graph supplies rules while the context graph supplies precedent. A versioned provenance store ensures that the complete graph state at the time of any decision can be reconstructed.

Example: the conflict detector queries the knowledge graph for the applicable separation standard (5 NM) and retrieves precedent data showing similar weather deviations in this sector were resolved by descent clearances.

Continuous Monitoring and Anticipation. Projection agents run continuously, not on demand. They model future states based on current trajectories, weather trends, and traffic flow, flagging potential conflicts before they become obvious. A task queue maintains emerging conflicts ranked by time-to-event and severity, pending advisories, and a watchlist of situations trending toward intervention. This supports aSA-1 (continuous perception) and aSA-3 (projection) as ongoing system functions rather than query-response interactions.

Example: the projection agent had already flagged converging trajectories between Aircraft A and B as a watchlist item. The pilot's deviation request escalates the situation from watchlist to active conflict assessment. The system anticipated the problem before the triggering event.

Multi-Agent Communication and Supervision. In a multi-agent system, the components can disagree. The conflict detector may assess "conflict in 4 minutes" while the projection agent's trajectory model indicates the conflict self-resolves. Inter-agent coherence (aSA-9) requires that such disagreements are detected and resolved before they reach the activation governor or the human operator. Passing unresolved contradictions to either would compromise aSA-8 and violate aSA-5.

Agent communication uses structured message passing rather than free-text exchange. Every inter-agent message carries a defined type, confidence score, source identifier, timestamp, and evidence references.

The supervisor agent implements both inter-agent coherence (aSA-9) and self-monitoring (aSA-10) as structured processes. For aSA-9, cross-agent consistency detection identifies contradictions between assessments and triggers re-evaluation, producing a reconciled operational picture before it reaches the activation governor or operator. For aSA-10, confidence calibration checks whether agents' confidence scores are consistent with cited evidence and historical calibration data, degradation detection monitors for stale data sources, anomalous agent outputs, or excessive processing latency, and competence boundary recognition flags situations outside the system's validated operating envelope.

When the supervisor identifies degraded or uncertain system SA, this propagates to the activation governor (shifting response behaviour toward conservatism) and to the presentation layer (signalling system confidence to the operator). The presentation layer also consumes the operator state model (aSA-6), derived from interaction patterns and workload proxies such as sector traffic count, calibrating alert volume and prominence to the operator's current capacity. A further consideration is epistemic independence. If all

agents share the same base model, their systematic errors are likely correlated, undermining the value of multi-agent cross-checking for aSA-9 and aSA-10. In safety-critical deployments, model heterogeneity may be required to ensure that redundancy provides genuine independence at the reasoning level.

Activation Governance in Practice. The activation governor (aSA-8) consumes three inputs: the situation assessment from reasoning agents, the supervisor's coherence and confidence evaluation (aSA-10), and authority rules encoded in the domain knowledge graph. For each situation, it determines the response class: inform (low severity), recommend (significant, time allows deliberation), recommend with urgency (significant and time-critical), or act with confirmation (critical, requiring immediate response with single-action acceptance). The ATC implementation caps the hierarchy at "act with confirmation", deliberately excluding fully autonomous action because the controller holds legal responsibility for separation assurance. This exclusion is itself a design decision encoded in the authority rules and auditable as such. More broadly, the architecture enforces a no-actuation principle: all agent interactions with the operational environment are read-only, and the system queries data but never issues commands to operational systems directly.

Example: the activation governor evaluates the conflict. The time horizon is short (four to five minutes) and the supervisor confirms cross-agent coherence with moderate-high confidence. The operator state model indicates moderate workload. The governor classifies this as recommend with urgency, presenting conflict assessment, resolution options, and precedent data with a visual urgency indicator.

Technical Failure Modes. The architecture has identifiable failure modes at each layer: orchestration failures (agent timeouts, deadlocks), knowledge failures (graph staleness, provenance corruption), agent-level failures (hallucinated outputs, poor calibration), and supervisor-level failures (missed degradation, undetected inter-agent contradictions). Each maps to specific aSA requirements. The broader concerns of automation complacency (Parasuraman and Manzey, 2010; Lee and See, 2004) are not resolved by architecture alone. Research in human reliability analysis suggests that below a certain threshold of AI diagnostic accuracy, automation bias effects may outweigh the advisory value, potentially degrading rather than improving operator performance. This places additional weight on reliable self-monitoring (aSA-10) and transparent communication of system confidence to the operator (aSA-5).

DISCUSSION AND CONCLUSION

This paper has formalised artificial situation awareness as ten architectural requirements derived from established SA theory. The comparative assessment suggests that neither foundation LLMs nor conventional RAG systems can satisfy these requirements, not because they are poor technologies, but because they were designed for different purposes.

The technical architecture reveals which requirements are readily met and which remain research challenges. Auditability (aSA-7) and transparency (aSA-5) emerge as structural properties of the orchestration pattern, since workflow engines inherently produce provenance chains and execution graphs. Inter-agent coherence (aSA-9) is partially addressed by supervisor consistency checking, but scaling to cross-domain deployments where agents operate under different ontologies introduces reconciliation problems that structured messaging alone does not resolve. Operator state modelling (aSA-6) depends on observable signals that vary by domain. The ATC simulation provides rich interaction data, but operational deployments may require biometric sensing. Self-monitoring (aSA-10) is the hardest requirement. Confidence calibration requires ground truth, available in simulation but difficult to establish in operations. Competence boundary recognition requires the system to know what it does not know, connecting to open questions in out-of-distribution detection.

Governed activation (aSA-8) depends on the same self-monitoring quality. The allocation of authority should be situation-dependent rather than fixed at design time. Parasuraman et al. (2000) distinguishes acquisition, analysis, decision, and action as four information-processing stages at which automation can be applied, each allowing multiple levels of automation.

The aSA framework maps onto this taxonomy. aSA-1 through aSA-3 address acquisition and analysis, while aSA-8 governs decision and action authority, which requires reliable confidence assessment. Operator state modelling (aSA-6) adds a further dimension, as the system's response should adapt both to the situation and to the operator's current capacity to act on it. A well-functioning system may itself become a source of complacency (Parasuraman and Manzey, 2010), and the architecture can mitigate this only partially.

A related challenge is response latency. Multi-agent processing introduces delays through API calls, graph queries, inter-agent messaging, and supervisor evaluation. In time-critical situations, these delays may render output operationally irrelevant. Hollnagel's (1993) contextual control model (COCOM) describes how human operators shift from strategic to tactical to opportunistic control modes as available time decreases, accepting reduced thoroughness for timely action. An analogous mechanism is needed. Therefore aSA-10 should track processing time against situation urgency and, when latency threatens acceptable bounds, trigger a mode shift from full deliberation to faster pathways. Candidate mechanisms include pre-compiled response templates, reduced-depth reasoning chains, and pre-computed decision boundaries, which are computational analogues to the "fast and frugal" heuristics of Gigerenzer and Todd (1999).

The aSA concept extends the distributed SA framework (Salmon et al., 2009). The requirements specify the AI system's contribution to joint human-AI SA: transparency provides accessibility, operator state modelling represents the human's perspective, and activation governance places the authority boundary within the shared situation picture. The framework is presented for a single domain, but aSA-9 is designed with cross-domain extension in mind, where agentic networks span organisational boundaries, reconciliation of domain-specific ontologies becomes a first-order design problem.

The aSA requirements discussed in this paper are proposed rather than validated, and further work is needed to determine their completeness and the extent to which aSA-6 and aSA-10 can be operationalised with current technology. The next phase is prototype implementation and evaluation measuring impact on user SA, workload, trust calibration, and decision quality.

REFERENCES

Chen, J.Y.C., Lakhmani, S.G., Stowers, K., Selkowitz, A.R., Wright, J.L., Barnes, M. (2018). "Situation awareness-based agent transparency and human-autonomy teaming effectiveness." *Theoretical Issues in Ergonomics Science*, vol. 19, no. 3, pp. 259–282.

Chen, L., Xu, J., Wu, T., Liu, J. (2024). "Information Extraction of Aviation Accident Causation Knowledge Graph: An LLM-Based Approach." Electronics, vol. 13, no. 19, 3936.

Edge, D. et al. (2024). "From Local to Global: A Graph RAG Approach to Query-Focused Summarization." arXiv preprint arXiv:2404.16130.

Endsley, M.R. (1995). "Toward a theory of situation awareness in dynamic systems." *Human Factors*, vol. 37, no. 1, pp. 32–64.

Endsley, M.R. (2017). "From here to autonomy: Lessons learned from human-automation research." *Human Factors*, vol. 59, no. 1, pp. 5–27.

Endsley, M.R. (2023). "Supporting human-AI teams: Transparency, explainability, and situation awareness." *Computers in Human Behavior*, vol. 140, 107574.

Gigerenzer, G., Todd, P.M., and the ABC Research Group (1999). Simple Heuristics That Make Us Smart. Oxford University Press, New York.

Gould, D.S.W., De Ath, G., Carvell, B., Pepper, N. (2025). "AirTrafficGen: Configurable Air Traffic Scenario Generation with Large Language Models." In Proc. NeurIPS 2025 Workshop on LAW.

Hildebrandt, M. (2026). "From Built-in Knowledge to Situated Understanding: A Framework for LLM Situation Awareness Using Real-Time Environmental Data." In *Proc. IEEE CogSIMA 2026*, in press.

Hollnagel, E. (1993). Human Reliability Analysis: Context and Control. Academic Press, London.

Jones, D.G., Endsley, M.R. (1996). "Sources of situation awareness errors in aviation." Aviation, Space, and Environmental Medicine, vol. 67, no. 6, pp. 507–512.

Lee, J.D., See, K.A. (2004). "Trust in automation: Designing for appropriate reliance." *Human Factors*, vol. 46, no. 1, pp. 50–80.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Kuttler, H., Lewis, M., Yih, W., Rocktaschel, T., Riedel, S., Kiela, D. (2020). "Retrieval-augmented generation for knowledge-intensive NLP tasks." In *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 9459–9474.

Parasuraman, R., Manzey, D.H. (2010). "Complacency and bias in human use of automation: An attentional integration." *Human Factors*, vol. 52, no. 3, pp. 381–410.

Parasuraman, R., Sheridan, T.B., Wickens, C.D. (2000). "A model for types and levels of human interaction with automation." *IEEE Transactions on Systems, Man, and Cybernetics: Part A*, vol. 30, no. 3, pp. 286–297.

Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., Zhang, Y., Tang, S. (2024). "Graph Retrieval-Augmented Generation: A Survey." arXiv preprint arXiv:2408.08921.

Salmon, P.M., Stanton, N.A., Walker, G.H., Jenkins, D.P. (2009). Distributed Situation Awareness: Theory, Measurement and Application to Teamwork. Ashgate.

Stanton, N.A., Stewart, R., Harris, D., Houghton, R.J., Baber, C., McMaster, R., Salmon, P., Hoyle, G., Walker, G., Young, M.S., Linsell, M., Dymott, R., Green, D. (2006). "Distributed situation awareness in dynamic systems: theoretical development and application of an ergonomics methodology." *Ergonomics*, vol. 49, no. 12–13, pp. 1288–1311.

Zhang, Z., Yang, X., Zeng, C. (2024). "A Domain-specific Retrieval-Augmented Generation System for Civil Aviation Safety." In Proc. Int. Conf. Big Data, Artificial Intelligence and Risk Management, ACM.