Human Interaction and Emerging Technologies (IHIET 2026), Vol. 236, 2026, 19–27
https://doi.org/10.54941/ahfe1008078

To Boldly Go Where AI Must Not Go Alone: Designing for Non-Delegable Human Authority in AI-Assisted Expert Work

Raimund Lehle[1] and Julia Kaesmayr[2]

[1]University of Applied Science – Public Administration and Finance, Ludwigsburg, Germany
[2]Carinthia University (CU) of Applied Sciences, School of Management, Villach, Austria

ABSTRACT

AI-assisted expert work raises a recurring question for human-machine systems design: where must human authority remain structurally located, rather than delegated to behavioural monitoring of the automated system? Conventional oversight is operationalised as post-hoc error detection, which the human factors literature shows fails under complacency and vigilance decrement at high reliability. We propose a design pattern that re-frames oversight as a structural property of the design process rather than a behavioural demand on the operator. We define Non-Delegable Junctures (NDJs): procedural stages whose constitutive quality criteria are human-authored and which resist delegation regardless of agent capability. For qualitative content analysis, we identify three NDJs – domain-knowledge curation, iterative codebook refinement, and communicative validation – and present an architecture enforcing them through codified process steps, immutable knowledge layers, and strict separation of a development agent from an application agent. The architecture suggests reproducible per-step audit artefacts without depending on reasoning-chain faithfulness. We anchor the pattern in the automation-monitoring tradition and link it to the EU AI Act's requirement for effective human oversight (Article 14). The contribution offers a transferable model for AI-assisted expert work in which human authority is enforced by design.

Keywords: Human-machine systems, Design process, Models and approaches, Non-delegable points, Human oversight, Audit trail

INTRODUCTION

Problem Setting

AI-assisted expert work is a fast-emerging mode of human-machine systems in which the human collaborator is not the end-user of a passive interface but the co-author of a methodologically structured procedure. As such systems become more capable, the question of where human authority must remain located ceases to be a question about user experience and becomes a question about the design process itself: at which procedural stages is human authorship constitutive, and at which is it merely conventional? The standard answer in the human factors literature operationalises oversight as

Received May 22, 2026; Revised June 17, 2026; Accepted July 1, 2026; Available online August 25, 2026

a behavioural performance – the operator monitors the agent, detects errors, and intervenes – and locates authority implicitly at runtime, in the vigilance of the operator. We argue that this answer is, by itself, structurally insufficient.

Gap

Decades of automation research indicate that operator vigilance degrades precisely under the high-reliability conditions that motivate the introduction of automation in the first place, and that the operator's perception of system reliability is itself shaped by the system under observation. Conceiving oversight as runtime monitoring therefore makes the safeguard dependent on the very factors that erode it. This is a structural rather than an incidental finding; it cannot be repaired by training, interface tuning, or further improvements to agent reliability. What is missing is a complementary design move: a way to express where in a procedure human authority must remain located, expressed not as a behavioural demand on the operator but as a structural property of the design itself.

Contribution

We propose such a design move. We define Non-Delegable Junctures (NDJ) as procedural stages whose constitutive quality criteria are human-authored and whose delegation would dissolve the criteria that define them. We identify three NDJs for qualitative content analysis – domain-knowledge curation, iterative codebook refinement, and communicative validation – and present a domain-agnostic architecture that enforces them through codified discrete process steps, immutable knowledge layers, and strict separation of a development agent from an application agent. We characterise the audit-trail properties this design entails and anchor the pattern in the automation-monitoring tradition and in the EU AI Act's requirement for effective human oversight. The remainder of the paper develops the concept (Section 3), the architecture (Section 4), its audit properties (Section 5), and their broader implications (Section 6).

BACKGROUND

Automation Monitoring and Appropriate Reliance

Two strands of the human factors literature frame the problem this paper addresses. The first concerns the conditions under which operators monitor highly reliable automation. Parasuraman and Manzey (2010) observed that complacency and automation bias are reliably elicited under exactly the high-reliability conditions that motivate automation in the first place, and that the resulting performance degradation reflect systematic interactions between attentional, situational, and automation-related factors rather than isolated operator errors. Vigilance decrement, well documented since the early monitoring literature and reiterated by Bainbridge (1983) in her account of the ironies of automation, exacerbates this pattern: as systems become more reliable, operators monitor them less actively and may become less able to detect or respond to failures. The second strand concerns how operator trust calibrates to system behaviour. Lee and See (2004) argued that

appropriate reliance depends on a correspondence between perceived and actual trustworthiness, but that system behaviour and feedback can in turn influence how operators perceive reliability. Taken together, the two strands indicate that monitoring-based oversight is conditioned on factors the system itself co-produces; treating it as an independent safeguard is therefore architecturally fragile.

Procedural Guidelines for Human-AI Interaction

Beyond the automation-monitoring tradition, recent work has approached human-AI interaction at the level of procedural guidelines. Amershi et al. (2019) consolidated eighteen guidelines for human-AI interaction, organised around the stages of use rather than around attitudinal constructs. The contribution we develop here is compatible with that procedural orientation: rather than asking how to make oversight attitudinally more effective, we ask how to position it structurally within the design process. Where Amershi et al. specify what an AI-assisted interface should do over time, we specify where in a methodologically defined procedure human authority must remain located.

Application Field

We use qualitative content analysis as the application field in which to identify Non-Delegable Points. The method is well suited to this purpose because its quality criteria – procedural documentation and intersubjective traceability – are explicit and methodologically codified, and because its constitutive activities (category-system construction, communicative validation) admit a clear distinction between agent-assistable and human-authored steps. We treat the method as a methodological setting, not as a theoretical commitment; the architecture proposed in Section 4 is domain-agnostic, and the three NDJs identified in Section 3 are presented as instantiations of a transferable pattern rather than as method-specific claims.

THE NON-DELEGABILITY-CONCEPT

From Monitoring to Structural Non-Delegability

The literature surveyed in Section 2 shares a common operationalisation: human oversight is conceived as a behavioural performance – the operator monitors, detects, intervenes – and the question of where authority resides is left to be answered at runtime. We propose a different operationalisation. A procedural step is *non-delegable* when its constitutive quality criteria are human-authored, that is, when the criteria that determine whether the step has been performed adequately are themselves expressions of human authority over the method. This is a constitutive position, not an empirical one: the question is not whether an agent could in principle approximate the step under current or future capability, but whether the step, once delegated, would still satisfy the criteria that define it. Framed this way, non-delegability is invariant under agent improvement and locates oversight in the design of the process rather than in the vigilance of the operator. We

call such procedural stages *Non-Delegable Junctures* (NDJs). The remainder of this section identifies three NDJs for qualitative content analysis; their architectural enforcement is the subject of Section 4.

NDJ1: Domain-Knowledge Curation

Before any agent skill consults a domain layer, the conceptual vocabulary of the analysed field must be assembled, scoped, and stamped as authoritative. This is documented domain authority, not automatable retrieval: the question of what counts as the relevant conceptual neighbourhood for a given study is a methodological judgement about proximity to the object of analysis, made by a human with disciplinary expertise. Delegating this step would replace authored scope with retrieved scope and dissolve the very criterion the step exists to satisfy.

NDJ2: Iterative Codebook Refinement

The construction of the category system is the hermeneutic core of qualitative content analysis. An agent may propose candidate categories, surface tensions in the emerging system, or flag segments that resist coding; but the controlling decision – accept, modify, reject – remains human, because the criterion of adequacy for a category is argumentative interpretive defensibility, not statistical convergence.

NDJ3: Communicative Validation

Once a category system is locked, its inter-subjective adequacy must be tested through structured human dialogue with subjects, peers, or other stakeholders in the analysed domain. This step is not an audit of the agent's behaviour but a methodological check on the category system itself; it cannot be performed by the agent because the agent is not a participant in the communicative community whose intersubjectivity is at stake.

ARCHITECTURE

We now describe an architecture that operationalises the three NDJs identified in Section 3 as structural properties of the design process, rather than as behavioural demands on the operator. The architecture is domain- and implementation-agnostic; we present it at the level of mechanism, not of tooling.

Three Design Principles

The architecture rests on three principles, each addressing a distinct failure mode of conventional human-in-the-loop arrangements.

First, **codified discrete process steps:** every methodological step is expressed as a named, parameterised procedure with an explicit precondition, action, and post-condition. Steps are not free-form prompts but skill-like units that the agent retrieves and executes from a curated catalogue. This rules out implicit method drift across runs and produces a stable substrate for per-step inspection.

Second, **immutable methodological and domain knowledge layers:** the procedural knowledge (what counts as a coding step, what a category requires) and the domain knowledge (the conceptual vocabulary of the analysed field) are held in versioned, read-only layers that the agent may query but not modify. Changes to either layer are authored by humans and produce a new version; runtime updates are excluded.

Third, **strict role separation between a development agent and an application agent:** the agent that participates in constructing the category system is not the agent that applies the system to material. The two roles are instantiated as distinct agents with separate contexts and no shared state at runtime.

A schematic of the architecture is shown in *Figure 1:* the three knowledge layers feed two role-separated agents whose interaction with the human is mediated by the codified skill catalogue, with NDJ checkpoints marking the procedural stages at which human authority is structurally located.

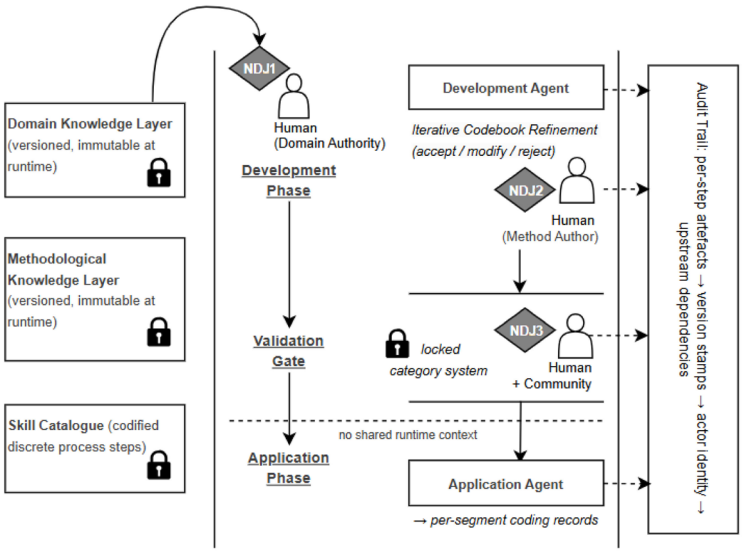

Figure 1: Architecture overview: knowledge layers, two-agent role separation, and NDJ checkpoints within the skill-based process.

Two-Agent Design With Strict Separation

The two-agent design serves a specific argumentative function. In qualitative content analysis, the construction of the category system and its application to material are methodologically distinct activities; conflating them in a single agent re-introduces, at runtime, the dependency that the design intends to avoid. By separating the development agent from the application agent and denying them shared context, the architecture approximates the methodological idea of independent coding without claiming to replicate inter-coder agreement statistically. The arrangement is also a structural analogue of confirmability in the sense used by Lincoln and Guba (1985): the audit trail of the application agent can be inspected without being

confounded by the category-construction process. We invoke this connection en passant; the architecture stands on its design-process rationale, not on a constructivist theoretical commitment.

Skill-Based Process and NDJ Checkpoints

The skill catalogue is the executable surface of the process. Each skill is a discrete procedure (e.g., *retrieve domain context*, *propose category for segment*, *apply category system to text unit*, *export coding record*), and skills are composed into a fixed process flow. NDJ checkpoints are explicit positions in this flow at which human authorship is required rather than optional. NDJ1 (domain-knowledge curation) is a *precondition* checkpoint: no agent skill that consults the domain layer may execute unless that layer carries a human-authored version stamp. NDJ2 (iterative codebook refinement) is located within the development agent's loop: the agent may propose category candidates, but the controlling decision – accept, modify, reject – is a human skill that writes back to the methodological layer. NDJ3 (communicative validation) is positioned between development and application: the application agent's run is gated on a documented human validation of the locked category system. The checkpoints are not advisory; they are structural preconditions, enforced by the skill orchestration rather than by operator vigilance.

Compliance Summary

Table 1 summarises how the three principles, the two-agent design, and the three NDJ checkpoints jointly cover the architectural commitments introduced above. The first column names the mechanism, the second the responsible actor, the third indicates whether the step is non-delegable, and the fourth names the audit artefact produced.

Table 1: Architectural compliance summary.

Mechanism	Actor	Non-Delegable	Audit Artefact
Domain knowledge curation	Human	Yes (NDJ1)	Versioned domain layer with author stamp
Skill catalogue invocation	Application agent	No	Per-step skill log
Category proposal	Development agent	No	Candidate record with provenance
Codebook refinement decision	Human	Yes (NDJ2)	Versioned methodological layer with rationale
Communicative validation	Human	Yes (NDJ3)	Validation record gating the application run
Coding application	Application agent	No	Per-segment coding record

The table is read row by row: each non-delegable row corresponds to one NDJ and to one human-authored artefact in the audit trail; each delegable row corresponds to an agent-produced artefact that is reproducible given the locked knowledge layers.

AUDIT TRAIL PROPERTIES

What the Architecture Structurally Enforces

The architecture described in Section 4 produces, by construction, a per-step audit record. Each invocation of a skill writes an artefact tied to the version stamp of the methodological and domain layers it consulted, to the identity of the responsible actor (human or one of the two agents), and to the upstream artefacts on which it depends. Because the knowledge layers are immutable at runtime, the audit record references a fixed substrate; because the skill catalogue is codified, the recorded steps are reconstructible without recourse to free-form prompt logs. Three properties follow. First, *traceability*: any coding decision in the application agent can be traced backwards through a chain of artefacts to the human-authored NDJ checkpoints that gated it. Second, *reproducibility at the design level*: holding the knowledge layers and the skill catalogue constant, an independent run reproduces the procedural structure, even where stochastic agent outputs differ. Third, *separability of human and agent contributions*: the audit trail distinguishes human-authored artefacts (versioned layers, NDJ decisions, validation records) from agent-produced artefacts (proposals, applications, exports), so post-hoc inspection can attribute responsibility without re-running the system. These properties are structural consequences of the design, not behavioural achievements of the operator; they hold under the same high-reliability conditions in which monitoring-based oversight is shown to degrade.

What the Architecture Does Not Resolve

The audit trail establishes traceability of process, not faithfulness of reasoning. Turpin et al. (2023) demonstrated that the chain-of-thought rationalisations produced by large language models can systematically misrepresent the factors that actually drove their outputs. Our architecture inherits this limitation: where an agent skill writes a justification, that justification is recorded but its faithfulness to internal computation is not warranted. We treat this as a design choice rather than a defect. The NDJ pattern is constructed so that the load-bearing instances of human authority – knowledge layers, codebook decisions, communicative validation – do not depend on agent rationalisation. Auditability of what the system did, under which *human-authored constraints*, and with which inputs, is preserved even if the agent's stated reasoning is unreliable. The pattern thus survives the Turpin condition by not requiring what Turpin shows cannot be guaranteed.

DISCUSSION

Relation to Human Factors and Human-Computer Interaction Research

The pattern proposed here complements the automation-monitoring tradition rather than replacing it. Where that tradition characterises operator-side limits of oversight under high automation reliability, our contribution provides a design-side response: positions in the procedure at which authority is structurally located in human-authored artefacts. Avril, Cegarra, Wioland and Navarro (2022) show that monitoring performance under variable automation reliability is shaped by contextual conditions that may not be fully accessible or controllable for operators in real time, which motivates the introduction of complementary structural mechanisms. The NDJ pattern offers one such mechanism. From an HCI perspective, the architecture instantiates the procedural orientation advocated by Amershi et al. (2019) at a coarser granularity: rather than guidelines applied within an interface, NDJs are stages within a methodologically defined process. The contribution is therefore best read as a model for design processes and models and approaches in human-machine systems, situated alongside – not against – existing guideline-based and monitoring-based approaches.

Regulatory Anchoring: EU AI Act Article 14

The European Union Artificial Intelligence Act (Regulation 2024/1689) requires, in Article 14, that high-risk AI systems be designed to enable effective human oversight by natural persons during the period in which the system is in use. The provision is operationalised at a high level of abstraction; concrete design mechanisms are left to providers and deployers. The NDJ pattern offers one operationalisation: by locating authority in immutable knowledge layers, NDJ checkpoints, and a separability-preserving audit trail, the architecture provides assignable structural positions for the oversight Article 14 requires, rather than relying on the operator's runtime vigilance. We do not claim that the pattern by itself constitutes compliance; we claim that it supplies a design template against which compliance arguments can be constructed.

Limitations and Scope

Three limitations qualify the contribution. First, the three NDJs identified here are specific to qualitative content analysis; identifying NDJs in another domain requires the same analysis of constitutive quality criteria and is not automatable. Second, the architecture has been characterised at the level of mechanism; the scalability of agent execution time and the integration with concrete agent frameworks remain to be examined. Third, we present no empirical validation of the pattern's effects on practitioners' work, and we make no empirical claim about it; the contribution is conceptual and architectural.

CONCLUSION

We have proposed Non-Delegable Junctures (NDJs) as a design-side complement to behaviour-based oversight in human-machine systems for AI-assisted expert work. By locating authority in immutable knowledge layers, codified process steps, and strict separation of a development agent from an application agent, the architecture positions oversight as a structural property of the design process rather than as a runtime demand on the operator. The pattern is presented as a transferable model for AI-assisted expert work and offers one design template against which compliance with effective-human-oversight requirements can be constructed. Identifying NDJs in further domains is itself a non-delegable task.

REFERENCES

Amershi, Saleema/Weld, Dan/Vorvoreanu, Mihaela/Fourney, Adam/Nushi, Besmira/Collisson, Penny/Suh, Jina/Iqbal, Shamsi/Bennett, Paul N./Inkpen, Kori. Guidelines for Human-AI Interaction. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, 1–13. Available online at https://dl.acm.org/doi/10.1145/3290605.3300233 (accessed 5/21/2026).

Avril, Eugénie/Cegarra, Julien/Wioland, Liên/Navarro, Jordan (2022). Automation Type and Reliability Impact on Visual Automation Monitoring and Human Performance. International Journal of Human–Computer Interaction 38 (1), 64–77. https://doi.org/10.1080/10447318.2021.1925435.

Bainbridge, Lisanne (1983). Ironies of automation. Automatica 19 (6), 775–779. https://doi.org/10.1016/0005-1098(83)90046-8.

European Parliament and Council of the European Union. (2024) Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Official Journal of the European Union, L 2024/1689. https://eur-lex.europa.eu/eli/reg/2024/1689/oj

Lee, John D./See, Katrina A. (2004). Trust in automation: designing for appropriate reliance. Human factors 46 (1), 50–80. https://doi.org/10.1518/hfes.46.1.50_30392.

Lincoln, Yvonna S./Guba, Egon G. (1982). Establishing Dependability and Confirmability in Naturalistic Inquiry Through an Audit. New York, 19.03.1982. Available online at https://eric.ed.gov/?id=ED216019.

Parasuraman, Raja/Manzey, Dietrich H. (2010). Complacency and bias in human use of automation: an attentional integration. Human factors 52 (3), 381–410. https://doi.org/10.1177/0018720810376055.

Turpin, Miles/Michael, Julian/Perez, Ethan/Bowman, Samuel R. (2023). Language Models Don't Always Say What They Think: Unfaithful Explanations in Chain-of-Thought Prompting. https://doi.org/10.48550/ARXIV.2305.04388.