International Journal of Computer (IJC)
ISSN 2307-4523 (Print & Online)
https://ijcjournal.org/index.php/InternationalJournalOfComputer/index

Evolution of the IT Project Manager's Role under Hyperautomation of Corporate Processes

Ievgenii Lysenko*

IT Project Manager, PwC, Vinnytsia, Ukraine
Email: yevhenii.lysenko@vntu.net

Abstract

The article examines how hyperautomation changes the role of the IT project manager in large corporate environments and explains the shift from manual coordination to event-driven process orchestration. The study addresses the growing complexity of distributed IT operations, stricter regulatory expectations, and the limits of managerial routines built on fragmented communication, manual checks, and dependence on individual actors. The article is designed as a conceptual paper grounded in literature on operational resilience, human error, distributed coordination, and project management, with SOAS used as an illustrative architecture rather than as a validated case study. The central contribution is a conceptual model of the IT orchestrator, a project leader who defines event logic, routing rules, validation layers, and escalation paths across interconnected systems. The analysis shows that hyperautomation changes the managerial object of work. Attention shifts toward process architecture, exception design, latent vulnerability mapping, and control of integration delay. A competency profile for the IT orchestrator is proposed. The article may be useful for researchers of digital governance, IT project managers, business process architects, and digital transformation leaders.

Keywords: hyperautomation; IT project management; operational resilience; integration delay.

Received: *2/24/2026*
Accepted: *4/24/2026*
Published: *5/1/2026*

---

* *Corresponding author*.

1.Introduction

The contemporary digital economy has entered a phase of complexity characterized by technological convergence and a radical transformation in the requirements for business operational reliability [1]. The management of distributed IT projects during multiterritorial expansion encounters the physical and cognitive limits of human resources [2]. In corporations operating across many jurisdictions with divergent tax, legal, and technical requirements, manual process coordination becomes a persistent source of vulnerability [3]. The relevance of this study is shaped by the gap between the speed of technological development and the inertia of managerial methods. Research on information systems projects shows that project success remains uneven and is strongly influenced by the quality of management processes, coordination routines, and execution discipline [4]. In distributed environments, this problem is intensified by human dependency, integration delay, and hidden time losses arising from data transfers between isolated information silos.

The research problem lies in the exhaustion of the possibilities of extensive growth of the administrative apparatus. Attempts to solve scaling problems by merely increasing the number of coordinators lead to a geometric rise in transaction costs and a degradation of management quality [5]. Within the operational resilience and critical infrastructure governance agenda reflected in current European regulatory frameworks such as NIS2 and related AI risk governance discussions, human error is treated as inseparable from system design, control logic, and accountability structures [6]. The purpose of the study is to explain how hyperautomation shifts the managerial center of gravity from task coordination to process orchestration in distributed corporate IT environments. The article's central contribution is a conceptual model of the IT orchestrator. This model describes a project leader who designs event logic, routing rules, validation mechanisms, and escalation paths within a resilient sociotechnical system.

2.Materials and Methods

This article is designed as a conceptual paper with an illustrative practice-informed model. It does not present a formal case study, comparative experiment, or statistical evaluation. The methodological aim is to synthesize literature on operational resilience, human error, distributed coordination, and project management, and to use that synthesis to interpret a concrete automation architecture.

The literature review covered publications from 2020 to 2025 indexed in Scopus, Web of Science, IEEE Xplore, and SpringerLink. The search combined the following thematic clusters: hyperautomation, event-driven architecture, operational resilience, project management, onboarding, validation, distributed teams, and human error. Sources were retained when they contributed to one of three analytical dimensions. The first dimension concerned latent vulnerabilities and resilience in complex systems [7–9]. The second concerned project performance, collaboration costs, and integration delay [4, 5, 10]. The third concerned the managerial implications of distributed cognitive load and information fragmentation [2, 3].

Internal technical documentation related to SOAS was used to reconstruct system modules, event flows, and validation logic. These materials were used for analytical illustration and architectural description. They were not

treated as independent evidence for causal claims about performance outcomes. The analytical procedure had three steps. First, the study identified recurring sources of fragility in distributed IT projects through the lenses of resilience engineering and error theory [7–9]. Second, it linked these vulnerabilities to managerial problems of coordination overload, schedule loss, and delayed value realization [2–5, 10]. Third, it mapped these problems onto the Smart Onboarding Automation System, abbreviated as SOAS, in order to illustrate how event-driven orchestration restructures managerial work. In this article, SOAS is treated as an internal conceptual architecture and practice-informed example. It is not used as independent empirical proof of causal performance effects. The contribution of the article lies in theory-informed conceptual clarification of role transformation under hyperautomation.

3.Results and Discussion

The discussion below develops a conceptual argument about role transformation. SOAS is used as an illustrative internal architecture that makes process logic, validation points, and escalation paths visible. The section does not claim formal causal proof of performance gains from a specific implementation. A substantial share of incidents in contemporary IT infrastructures that involve data integrity breaches or missed deadlines can be linked to human interaction with fragmented processes and tools [8]. Academic analysis requires attention to systemic vulnerabilities rather than isolated operator failure. According to James Reason's model, active failures, that is, operator errors, become critical only in the presence of latent conditions, such as hidden defects in process organization [9]. In distributed teams, latent conditions manifest themselves through information asymmetry and communication ruptures. The use of disconnected tools such as email, Slack, and Excel creates an organizational silo, where knowledge remains locked within departments. Table 1 presents a classification of the principal types of anthropogenic risks generating systemic instability in IT projects.

Table 1: Classification of anthropogenic operational risks in distributed IT projects

Parameter	Communication Breakdowns	Data Entry Errors	Validation Delays
Generation Mechanism	Fragmentation of information across time zones	Mechanical typos during manual transfer between systems	Process downtime due to reliance on a human-in-the-loop
Latent Condition, per Reason	Lack of a single source of truth, SSoT	Outdated interfaces, lack of API synchronization	Critical path depends on availability of a specific manager
Manifestation of Functional Resonance, per Hollnagel	Asynchronous workflows force specialists to infer requirements, leading to errors at the testing stage	Cognitive overload causes accumulation of micro-deviations, undermining global data integrity	Bottlenecks from a single employee's absence, vacation or sick leave, disrupt the value chain
Economic Impact	Departmental isolation, duplicated work, productivity loss	Tax penalties, distorted predictive analytics	Direct losses from idle high-cost resources, Cost of Delay

The problem of manual management becomes more acute as scaling proceeds. As the number of jurisdictions, systems, and approval interfaces grows, coordination links expand at a nonlinear rate and place additional cognitive load on administrative personnel [2, 3, 5]. Under such conditions, added layers of control may increase integration delay and transactional overhead instead of improving reliability [5].

Integration delay, in a managerial sense, constitutes transactional friction between the emergence of a need for resource synchronization and the moment at which value extraction from those resources actually begins. In traditional IT projects, onboarding may take from several days to several weeks [10]. This is a time during which the company pays wages but receives no product increment.

To quantify the damage, the Cost of Delay framework is applied. The cost of delay is defined as the partial derivative of expected financial value with respect to time. In the context of an IT project, the formula for aggregate losses takes the following form:

$$L_{Total} = \int_0^{T_{delay}} (C_{direct}(t) + C_{overhead}(t) + CoD(t) \times P_{contribution})dt + C_{retention}$$

where L_{Total} is total losses, C_{direct} is direct costs of maintaining an idle specialist, $C_{overhead}$ is operational overhead costs for maintaining the workstation, T_{delay} is integration delay time, CoD is lost profits from delayed release of a feature or product to market, $P_{contribution}$ is the contribution rate of this resource to the creation of final value, and $C_{retention}$ is the cost of replacing an employee. From this perspective, reduction of integration delay becomes a relevant managerial objective during global expansion.

One response to the crisis of manageability is hyperautomation, understood here as the integration of event-driven systems, validation logic, and automated decision routes within a shared process architecture. To illustrate this shift, the article uses the Smart Onboarding Automation System, abbreviated as SOAS, as an internal conceptual model of onboarding orchestration. In the present study, SOAS serves as an analytical example that helps specify managerial functions, control points, and exception paths. It does not serve as stand-alone empirical proof of the broader claims advanced in the article.

The SOAS architecture is based on Event-Driven Architecture. Instead of monolithic requests, the system reacts to triggers, that is, changes of state in primary HR systems. As soon as a candidate signs the offer, the event employee_hired is generated, launching a cascade of automations without human involvement.

The generated events are delivered to the orchestration layer, where the SOAS Orchestrator routes them, validates the context, and invokes the relevant domain modules. From an architectural perspective, the SOAS platform can be described as a set of services connected through an event broker and a shared rule base. The orchestrator listens for the employee_hired event. Once this event is received, the Identity and Access Management Module creates a default access profile based on role, department, and privilege level. In parallel, the Workplace Provisioning Module prepares the user's digital workplace, creates corporate accounts, and provisions core IT services. The Learning and Compliance Module assigns mandatory courses, policies, and regulations for review. The Social Integration Module assigns a mentor and connects the user to relevant communication channels, thereby creating

a structured welcome scenario. This division into event domains supports localized service responsibility and scaling of individual system segments without redesign of the full architecture.Above the operational contour lies an analytical and control layer that includes the Monitoring & SLA Control Module and the Anomaly Detection Engine. This level tracks the passage of each onboarding step, compares the actual state of the process with the target adaptation model, and identifies deviations: delays in granting access, failure to complete mandatory training, and the absence of a confirmed meeting with a manager or mentor. When an anomaly is detected, the system automatically launches an escalation mechanism through the Exception Handling Service, sending notifications to the responsible manager, HR partner, or IT administrator. From an analytical perspective, the SOAS architecture can be interpreted as a self-monitoring solution. It combines reactive event processing, modular decomposition of functions, and embedded feedback, which makes resilient and controllable onboarding execution conceptually possible under conditions of high organizational complexity. Figure 1 shows the conceptual architecture of the event-driven SOAS system.

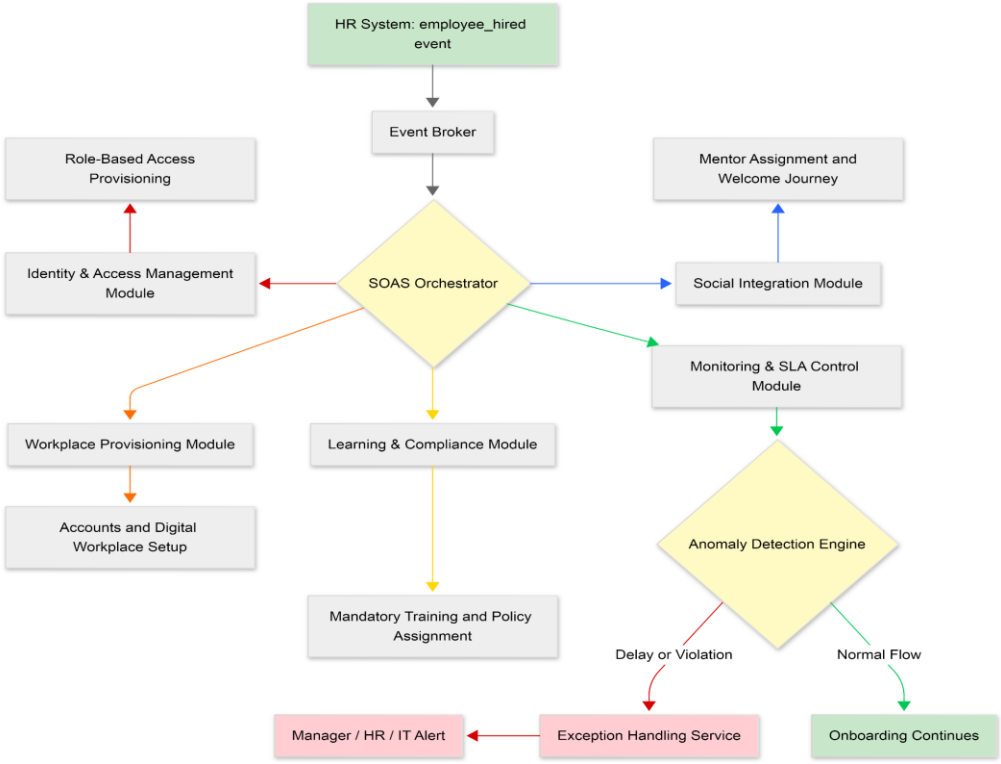

Figure 1: Conceptual architecture of an event-driven SOAS system

Within this conceptual architecture, the model of Compliance, Clarification, Culture, Connection, and Confidence is translated into explicit process rules. The system design includes automated mentor assignment, scheduling of introductory meetings through Graph API, and tracking of mandatory information security training as observable process states. Automation of primary resource input inevitably requires automation of data quality control. In multiterritorial IT projects, a critical point becomes cross-country validation of tax and financial information. Manual audit across many jurisdictions creates delay, uneven review quality, and escalation overload. To address this problem, the article uses a two-level validation model as a conceptual design pattern for cross-country data control.

The first level is structural automatic validation. All data entering from local offices passes through an algorithmic filter. The system checks syntax, data types, the presence of mandatory attributes, and compliance with schemas (JSON/XML). At this stage, a large share of routine format, completeness, and schema errors can be intercepted before expert review.

At the second level, functional expert validation is implemented. Only transactions that have passed the first level but contain logical anomalies, for example, a sharp deviation of the tax rate from the historical average, are forwarded to a human. This implements the human-in-the-loop principle, in which expert intelligence is used to analyze complex cases.

Hyperautomation changes the very object of management in an IT project. Previously, the manager held the process together through constant approvals, manual checks, and personal supervision of bottlenecks. Now the center of gravity shifts toward designing an organizational environment in which most standard decisions are made according to predetermined logic. It follows that the contemporary project leader is occupied less and less with distributing assignments and is increasingly responsible for constructing an integrated framework of work, data, and rules for responding to deviations. The value of this role is determined by the quality of the managerial architecture created. Such a shift is especially evident in distributed teams, where the principal threats arise from information discontinuities, confirmation delays, and multiple local interpretations of the same process. When information is dispersed across different channels, the manager becomes a living transmission node through which critical decisions flow. Hyperautomation can reduce this dependency by establishing a single source of reliable data and linking participants' actions through event logic. Under these conditions, the manager's role shifts toward semantic synchronization within the system. It is necessary to determine which event initiates the process, which states are considered normal, which deviations require intervention, and under which rules escalation should occur.

SOAS illustrates this transition at the level of process design. In this model, onboarding, access provisioning, training assignment, and inclusion in work communications are linked through one event-driven scenario. After the initiating HR event occurs, the system launches a coordinated sequence of actions across modules. The manager's work shifts toward defining routing rules, responsibility boundaries, state transitions, and exception thresholds. This architecture makes the move from task coordination to process orchestration visible in operational terms. The approach to risks also changes. Under the manual management model, the manager most often confronts the consequences of accumulated errors after deadlines have already been missed, data have been distorted, and project participants are occupied with searching for the guilty party. With the use of SOAS and similar solutions, the focus shifts to the hidden conditions from which incidents later emerge. Therefore, the manager of a new type of work addresses the causes of systemic fragility. Such a manager identifies areas where a decision depends on one person, where data are transferred manually, where verification begins too late, and where process observability is absent. This logic places latent vulnerability mapping at the center of managerial attention. In a hyperautomated environment, the project leader is expected to review such vulnerabilities during each scaling effort and each integration of a new country, function, or contractor.

Time management acquires special significance as an economic category. In a complex corporate environment,

delays cease to be mere organizational inconveniences and become a direct source of losses. As long as access has not been granted, training has not been assigned, and equipment has not been prepared, the company bears costs without creating a useful result. In conceptual terms, the SOAS model illustrates how an HR event can be linked to the immediate launch of dependent operations, thereby reducing the interval between hiring and the specialist's actual inclusion in productive work. Consequently, managerial analysis shifts toward the speed at which resources are converted into value. This perspective implies measurement of the duration of each transition between process states, recording of the cost of delay at key stages, and prioritization of automation in segments where idle time produces the greatest financial loss.

New competencies move to the foreground. It is no longer sufficient for a manager to know how to draw up a schedule, conduct meetings, and monitor execution. Skills are required in process description, event architecture, integration logic, data quality, and embedded control principles. Work with the SOAS model requires the ability to think in terms of states, transition rules, and domains of responsibility, since it is through such decomposition that scalability and resilience are achieved. In competency terms, this shift increases the importance of process route modeling, interface interaction between systems, methods of rule formalization, and the interpretation of analytical dashboards. Without this, there is a risk of remaining an administrator of someone else's automation.

Managerial responsibility related to transparency and procedural fairness also increases. When a system based on the SOAS model distributes access, assigns mandatory actions, identifies anomalies, and launches escalation, the manager becomes responsible for the fairness and interpretability of the process. This role requires understanding the grounds on which the algorithm produces a decision, the points at which bias may arise, the types of error that may be delegated to machine processing, and the cases that require expert participation. Within this logic, the two-level verification principle defines the boundary between formal control and contextual judgment. Routine determinate operations are assigned to the algorithm. Context-dependent and high-risk cases are escalated to a human reviewer. This model therefore relies on prior separation of operations by level of certainty and on explicit documentation of the boundaries of automatic decision-making.

Ultimately, the IT project manager's role shifts toward metagovernance and transitions into the IT orchestrator role. The SOAS model suggests that project resilience can be strengthened through a predesigned system of events, rules, checks, and feedback. The effectiveness of the manager manifests itself in how well processes recover after failure, how quickly deviations are detected, and how rarely the project requires manual rescue. This role can be described through several design priorities. These include specification of critical events and expected process reactions, creation of a unified and reliable data environment, automation of routine actions, integration of observability mechanisms, and definition of escalation rules. The remaining functions that depend on personal memory, correspondence, and improvised coordination indicate the residual boundary of manual management and the next area of professional transformation. A comparison of traditional and hyperautomated IT project management roles is shown in Table 2.

Table 2: Comparison of Traditional and Hyperautomated IT Orchestrator Roles

Role Dimension	Traditional IT Project Manager	IT Orchestrator under Hyperautomation	Practical Implication
Object of management	Manual coordination of tasks, deadlines, and performers	Design of process logic, event flows, and response rules	Managerial effort shifts toward process architecture
Information management	Retrieval of data from fragmented communication channels	Unified data environment and event-based synchronization	A single authoritative data environment becomes essential
Work initiation logic	Sequential manual actions across multiple functions	Automated process launch triggered by an employment event within an event-driven architecture	Integration delay should be minimized at the moment of process initiation
Risk management	Reaction to already manifested failures	Identification of latent conditions and structural vulnerabilities	The focus moves toward root causes of instability
Deviation control	Manual status tracking and follow-up	Continuous monitoring, anomaly detection, and escalation logic	Response mechanisms should be predefined within the process design
Competency profile	Planning, supervision, and administrative coordination	Process modeling, system integration, and data quality governance	Systemic thinking and process design capabilities become critical
Strategic role	Operational supervision of execution	Meta-governance and maintenance of process resilience	The manager's value is increasingly defined by systemic design capacity

This role transition is consistent with the literature reviewed in the article. Research on distributed cognition and information overload explains why coordination burdens intensify in dispersed environments [2, 3]. Studies of project outcomes and collaboration costs clarify why growth in coordination interfaces produces delay, waste, and reduced predictability [4, 5]. Resilience engineering and human error research explain why managerial attention shifts toward conditions, feedback loops, and detection points within the system [7–9]. On this basis, the IT orchestrator is proposed as a conceptual synthesis of these strands rather than as a formally established occupational category.

Table 2 summarizes the central analytical claim of this article. Under hyperautomation, the managerial center of gravity moves toward process architecture, event definition, data governance, and exception design. Communication, prioritization, and stakeholder alignment remain part of the role, yet their function changes within an automated environment. The term IT orchestrator is used here as a conceptual label for this reconfigured bundle of responsibilities.

This study has clear scope conditions. It does not report a formal case study, a comparative implementation

analysis, or a quantified performance evaluation of SOAS. For that reason, the article does not claim direct causal proof that a specific architecture will produce a given reduction in delay, error, or cost. Its contribution lies in theory-informed conceptual clarification. Future research can test the proposed model through longitudinal case studies, comparative onboarding metrics, and process-level measurement of delay, exception frequency, and recovery time.

4.Conclusion

This article argues that hyperautomation changes the role of the IT project manager by shifting the managerial object from manual coordination to process orchestration. In distributed corporate settings, rising system complexity, fragmented information flows, and regulatory pressure expose the limits of coordination routines built around human follow-up. The analysis therefore centers on event logic, validation design, exception handling, and observability as core elements of managerial work under hyperautomation.

Within this argument, SOAS functions as an illustrative architecture that makes the role transition analytically visible. Its value in this paper lies in showing how onboarding can be represented as a sequence of states, triggers, rules, and escalation paths. On that basis, the figure of the IT orchestrator is proposed as a conceptual model of a project leader who governs process architecture, latent vulnerabilities, and the conversion of organizational events into controlled execution.

The article does not claim empirical validation of SOAS as a universally proven solution. Its contribution lies in clarifying a role transformation and in providing a structured vocabulary for further empirical research. Future studies may test this model through comparative case designs, performance metrics, and cross-organizational analysis of automated process environments.

References

[1] Y. Fan and X. Shen, “Research Regarding the Impact Mechanism of Digital Economy Development on Economic Resilience—Mediating Effect Based on Upgraded Industries,” Sustainability, vol. 17, no. 13, Art. no. 5749, 2025. doi: 10.3390/su17135749.

[2] R. M. Jacobsen, J. Wester, H. B. Djernæs, and van Berkel, “Distributed Cognition for AI-supported Remote Operations: Challenges and Research Directions,” ArXiv, 2025. doi: 10.48550/arXiv.2504.14996.

[3] M. Arnold, M. Goldschmitt, and T. Rigotti, “Dealing with Information Overload: A Comprehensive Review,” Frontiers in Psychology Organizational Psychology, vol. 14, Art. no. 1122200, 2023. doi: 10.3389/fpsyg.2023.1122200.

[4] J. Varajao, R. P. Marques, and A. Trigo, “Project Management Processes – Impact on the Success of Information Systems Projects,” Informatica, vol. 33, no. 2, pp. 421-436, 2022. doi: 10.15388/22-INFOR488.

[5] R. Vivona, M. A. Demircioglu, and D. B. Audretsch, “The costs of collaborative innovation,” The Journal of Technology Transfer, vol. 48, pp. 873-899, 2022. doi: 10.1007/s10961-022-09933-1.

[6] N. Parlov, G. Akrap, and J. Esterhajer, “Supply Chain Security and AI Risk Governance Model for Critical Infrastructure under NIS2, CER, and CRA,” Applied Cybersecurity & Internet Governance, vol. 4, no. 1, 2025. doi: 10.60097/ACIG/211823.

[7] M. Fargnoli, L. Murgianu, and M. Tronci, “The Functional Resonance Analysis Method (FRAM) Application in the Healthcare Sector: Lessons Learned from Two Case Studies on Medical Device Management,” Applied Sciences, vol. 14, no. 20, Art. no. 9495, 2024. doi: 10.3390/app14209495.

[8] S. R. Grandhi and J. D. Still, “Deciphering Human Error: Improving Cybersecurity Reporting,” Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 69, no. 1, pp. 234-239, 2025. doi: 10.1177/10711813251358790.

[9] D. A. Wiegmann, L. J. Wood, T. N. Cohen, and S. A. Shappell, “Understanding the Swiss cheese model and its application to patient safety,” Journal of Patient Safety, vol. 18, no. 2, pp. 119-123, 2022. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8514562/

[10] T. Ye. Gura, O. O. Gura, L. A. Chernikova, and O. I. Gura, “Peculiarities of Onboarding in Modern IT Projects,” Information Technologies and Learning Tools, vol. 87, no. 1, pp. 357-374, 2022. doi: 10.33407/itlt.v87i1.4682.