International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Approaches to Incident Management and SAP S/4HANA Stabilization within Application Management

Rana Sudeepta[1]

[1]Senior Application Consultant, Tampa, USA

Abstract: ***This article examines how incident management should be rethought for SAP S/4HANA stabilization within application management, with particular attention to order-to-cash landscapes, cross-module dependencies, and post-go-live operational volatility. The topic is relevant to regulated, transaction-intensive environments where ERP incidents affect billing, logistics execution, compliance, and finance simultaneously. The study treats stabilization as an operational design problem within application management. The source base comprises 10 recent publications and industry materials covering ERP post-implementation support, IT service management, DevOps governance, monitoring, SAP process intelligence, and AI-enabled incident handling. The analytical part identifies three converging lines of effective stabilization: structured AMS governance, multi-layer observability tied to business processes, and knowledge-based incident routing with feedback into problem management. The article proposes an implementation logic, a monitoring model, and a decision framework for enterprise SAP support organizations after migration or a major transformation.***

Keywords: SAP S/4HANA, application management, incident management, stabilization, AMS, ITSM, DevOps, order-to-cash, observability, process mining

1. Introduction

SAP S/4HANA programs begin their use from the critical stage of operation phase, which increases their productivity. The period that follows often determines whether architectural intent is converted into reliable daily execution, or whether the landscape settles into recurring exceptions, unstable interfaces, manual workarounds, and overloaded support queues. The issue becomes acute in regulated and high-volume sectors such as life sciences, pharmaceuticals, healthcare, and global retail, where a disturbance in order capture, pricing, billing, transport coordination, or tax determination spreads across logistics, compliance, and finance.

The article aims to define analytically grounded approaches to incident management and SAP S/4HANA stabilization within application management.

The first objective is to identify how incident management in an SAP S/4HANA environment departs from generic ticket handling and turns into a structured stabilization function. The second objective is to determine which operational capabilities sustain stabilization after migration, conversion, or a major process redesign. The third objective is to formulate a practical implementation logic for application management teams responsible for L2 and L3 support, root-cause analysis, monitoring, and continuous improvement.

The working hypothesis of the study is that SAP S/4HANA stabilization within application management becomes more effective when incident management is organized as an integrated operational control model that links business-process observability, technical diagnostics, structured routing, root-cause analysis, and feedback into problem and change management.

The novelty of the article lies in connecting three strands that are often discussed separately: ERP post-implementation support, IT service management, and DevOps governance, as well as SAP-specific operational intelligence built on monitoring and transaction-level process data. This connection enables treating stabilization as a managed operating model with explicit inputs, routing rules, learning loops, and measurable outcomes.

2. Materials and Methods

The source corpus includes ten recent works selected for direct relevance to ERP post-implementation support, IT incident handling, DevOps-oriented operations, SAP process intelligence, and SAP S/4HANA operational tooling. The screening logic favored peer-reviewed studies that illuminate stabilization after implementation, service governance, monitoring design, and intelligent incident handling, while limiting non-academic material to one industry benchmark report and one official SAP operational framework document. Conceptually, the corpus covers five clusters: AMS in ERP post-implementation work [4], ERP sustainability and post-go-live failure patterns [8], ITSM governance and organizational control logic [7], DevOps success factors and monitoring capabilities [2], [5], transaction-level SAP data extraction for process-level diagnostics [3], and AI-supported incident routing and resolution [1], [9]. The SAP S/4HANA-specific operational layer is represented by a migration benchmark report and SAP Cloud ALM operations documentation, which ground the discussion in current transformation pressure and available observability mechanisms [6], [10].

The study relies on comparative analysis, source analysis, conceptual synthesis, typologization, and analytical generalization.

This study uses comparative analysis to align incident management, AMS, ITSM, and SAP-specific operational practices. Conceptual synthesis links separate literature streams into a stabilization model. Typologization distinguishes reactive support, controlled stabilization, and intelligence-guided optimization. Analytical generalization derives a practical operating logic for enterprise application management teams.

3. Results

Recent ERP literature treats post-implementation support as the stage at which routine business pressure tests the system's economic and operational viability. Bradford, Bucy, and Adivar connect ERP maintenance and support to the system's long-term viability and show that AMS arrangements differ depending on whether they are built for simple continuity or for value creation [4]. Mahmood and co-authors reach a related conclusion from post-ERP case studies, where sustainability after deployment depends on training, testing, personnel retention, and explicit post-implementation support [8]. MacLean and Titah add a broader management lens by treating ITSM as a control system oriented toward performance, transparency, and customer focus inside the IT function [7]. In SAP S/4HANA stabilization, incident management protects process continuity, controls recurrence, and converts operational noise into guided learning.

Transformation pressure often compresses the time available for operational maturation. The SAPinsider benchmark report notes that the approaching end of maintenance for core SAP Business Suite applications had become the biggest factor shaping S/4HANA plans among surveyed organizations, overtaking earlier external pressures and pushing many firms into evaluation, pilots, or active implementation [6]. Under such conditions, migration readiness and post-go-live stability become linked. A technically live landscape may still require a stabilization period before routine operations become reliable. For application management, this means that stabilization starts before incident volumes spike. It starts when the support model is designed, when ownership across OTC, FI, logistics, TM, WM, tax, and external carriers is assigned, and when documentation is written for diagnosis, ownership transfer, and support execution [4], [6].

A productive tension appears in the literature between service governance and operational agility. On one side, AMS success is associated with documentation quality, process standardization, and thoughtful treatment of customization during implementation [4]. On the other hand, DevOps research identifies organizational, technical, and socio-cultural success factors such as shared responsibility, automation, collaboration, and management support [2]. MacLean and Titah help reconcile the two positions by framing ITSM as a control system that preserves operational flexibility [7]. In SAP S/4HANA stabilization, the implication is precise. Standardization is needed for ownership, escalation, and auditability. Agility is needed for rapid routing, selective automation, and cross-team correction when issues travel across modules and interfaces. Incident management struggles when one of these poles dominates. Bureaucratic escalation models slow response, while improvised support models increase inconsistency and repeated failure.

Evidence from monitoring studies supports this conclusion. Giamattei and colleagues, after reviewing seventy-one monitoring tools in DevOps and microservice settings, show that monitoring capability requires multiple layers because tools differ in assumptions, observable signals, and analytical depth [5]. SAP's own operational design for Cloud ALM follows the same multi-layer logic. Its functional overview combines business process monitoring, integration and exception monitoring, user monitoring, job and automation monitoring, configuration and security analysis, health monitoring, event processing, alert management, and downstream problem routing and resolution [10]. For S/4HANA stabilization within application management, the consequences are clear. Useful observability goes beyond CPU, memory, and technical dumps. It must connect business KPIs, transactional failure points, interfaces, jobs, and alerting logic into a single operational chain [5], [10].

In practical SAP support work, this operational chain is anchored in concrete diagnostic instruments. ST22 helps isolate ABAP runtime dumps that reveal termination points in custom code, enhancements, forms, or standard processing. SM37 exposes failed or delayed background jobs, which is especially relevant for billing runs, output processing, settlement routines, and periodic integration tasks. SMQ1 and SMQ2 make queue-level errors visible in outbound and inbound asynchronous communication, allowing support teams to trace failures in qRFC-based data transfer, middleware-dependent flows, and cross-system message backlog. Used together, these transactions connect technical symptoms with process interruption points, thereby improving the precision of triage, escalation, and root-cause localization [10].

This relationship is summarized in Figure 1.

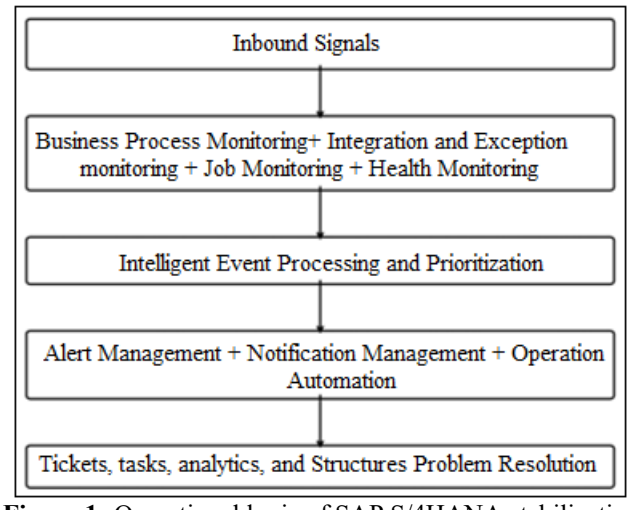

Figure 1: Operational logic of SAP S/4HANA stabilization within application management (adapted from SAP Cloud ALM for Operations functional overview [10])

Figure 1 presents an operational sequence that moves attention from isolated incident records to signal capture, interpretation, routing, and corrective action. Stabilization fails when this sequence is broken. Typical breakpoints include weak signal

capture, fragmented ownership between functional and technical teams, manual triage without business priority logic, and missing feedback from resolved tickets into reusable knowledge objects.

Berti and co-authors address one of the most expensive analytical steps in SAP environments, namely the extraction of event data from ERP tables for process mining [3]. Recurring incidents in order-to-cash processes rarely occur in isolation. A billing block may originate from pricing inconsistencies, incomplete master data, credit status issues, failed output processing, EDI mismatches, or transport-induced configuration drift. Object-centric event data extraction offers a way to reconstruct how these dependencies unfold across documents and tables without forcing diagnosis into one oversimplified case notion [3]. In application management terms, this creates a bridge between incident symptoms and process behavior. The support organization can identify stable recurrence patterns across sales order creation, delivery, shipment integration, billing, and financial posting.
The literature on intelligent incident handling strengthens the same line of argument from a different direction. Ahmed and colleagues present a knowledge-based system that automates two central components of incident service management, ticket assignment, group selection, and incident resolution, using a large real-world dataset of incident descriptions [1]. Peralta and co-authors go further by combining NLP, knowledge engineering, and mathematical resource allocation within an enterprise incident management framework built for scalability and integration with existing systems [9]. These two studies increase the need for expert functional architecture. In SAP S/4HANA, automation without a domain structure merely accelerates noise. Yet when classification models, knowledge artifacts, and routing rules are aligned with business objects, interface topology, and support ownership, incident handling becomes faster and more consistent [1], [9].

Ahmed et al. focus on rapid assignment and immediate resolution logic at the incident level [1]. Peralta et al. extend the frame to intelligent orchestration and resource allocation [9]. Berti et al. supply the process-data foundation needed to interpret recurrence and sequence across SAP transactions [3]. SAP Cloud ALM provides the operational shell in which alerts, monitoring domains, and problem routing can be unified [10]. Taken together, these works suggest that mature stabilization requires three linked capabilities. The first is diagnostic visibility into business execution. The second is structured incident routing and prioritization. The third is organizational learning that feeds resolved issues back into monitoring rules, knowledge articles, and change decisions. Mature application management requires diagnostic visibility, structured routing, and organizational learning.

The post-ERP literature explains why that reactivity persists so often. Mahmood and co-authors identify post-implementation support, system adoption, testing, and employee retention among the factors that shape sustainability after deployment [8]. Bradford, Bucy, and Adivar show that documentation and standardization practices established during implementation influence both transition difficulty and later AMS quality [4]. MacLean and Titah, in turn, locate the broader organizational effect of service management in transparency and customer focus [7]. These findings converge on a practical conclusion for SAP S/4HANA stabilization. Incident management works best when it is connected to problem management, release discipline, documentation governance, and process ownership from the start. Well-designed application management reduces recurrence, preserves design memory, and supports controlled optimization.

4. Discussion

The literature places stabilization between the transformation strategy and conventional support as a distinct operating interval. In an SAP S/4HANA setting, stabilization should be treated as the operating interval in which the architecture is forced to prove itself under production variability. For OTC-heavy environments, that interval is where pricing logic, credit control, logistics execution, billing output, EDI or IDoc flows, and financial integration either settle into a reliable pattern or begin to generate persistent friction. Application management becomes effective when it is explicitly organized around this interval.The team operates as a control layer linking business process stability, technical observability, routing discipline, and knowledge accumulation.

To make that control layer operational, it is useful to distinguish three models of AMS behavior. Table 1 compares them because many organizations still limit stabilization to queue movement.

Table 1: Operating models of SAP S/4HANA application management during stabilization

Operating model	Primary aim	Typical trigger logic	Dominant support behavior	Best-fit phase	Main weakness
Reactive ticket factory	Restore service quickly	Ticket arrival and urgency	Manual triage, local fixes, escalation by noise level	First weeks after go-live or crisis periods	Recurrence remains invisible
Controlled stabilization model	Reduce repeat failure and process volatility	Incident recurrence, business criticality, failed jobs, and interface exceptions	Structured ownership, RCA, ticket-to-problem linkage, documentation updates, monitored thresholds	Early to mid stabilization window	Progress slows if data quality is poor
Intelligence-guided optimization model	Suppress disruption before business impact expands	Trend shifts, process drift, anomaly patterns, predictive risk signals	Automated routing, knowledge reuse, selective automation, process mining, and feedback into change decisions	Mature support organization with stable governance	High dependence on disciplined data capture

The transition should move from reactive restoration to controlled stabilization and then to intelligent optimization. In practice, many SAP support functions fail because they attempt automation before they standardize ownership, incident taxonomy, and diagnostic evidence. That sequence produces fast routing without dependable resolution. In

regulated sectors, the damage is amplified because unresolved recurrence creates both service instability and traceability problems.

A second design question concerns what exactly should be measured during stabilization. Table 2 proposes a compact metric architecture. The metric architecture separates apparent workload from actual instability and shows whether the support organization is learning.

Table 2: Monitoring metrics for SAP S/4HANA stabilization within application management

Stabilization dimension	Metric	Operational meaning	Review rhythm
Service restoration	Mean time to acknowledge, mean time to resolve, and reopened incident rate	Indicates queue responsiveness and quality of closure	Daily and weekly
Business process continuity	Blocked sales orders, failed billing jobs, delivery exception recurrence, overdue interfaces	Shows whether the OTC flow is stable in production	Daily and weekly
Landscape reliability	Failed background jobs, alert noise ratio, data collection health, exception burst frequency	Distinguishes real risk from telemetry overload	Daily
Knowledge maturity	RCA completion rate, reusable solution article rate, repeated incident share	Measures whether support work is accumulating reusable intelligence	Weekly
Change discipline	Emergency change share, transport rollback frequency, and post-release defect leakage	Reveals whether stabilization is being undermined by uncontrolled change	Weekly and monthly
Ownership clarity	Cross-team handoff count, unresolved dependency age, and module-level backlog aging	Exposes friction between functional, technical, and integration teams	Weekly

These metrics matter because stabilization is often misread through one number, usually ticket volume. System improvement is tracked through recurrence, workaround normalization, monitoring depth, and post-change leakage. A healthy stabilization period may initially generate more documented work because hidden dependencies become visible. A weak stabilization period may show fewer incidents simply because users have normalized manual workaround behavior or because monitoring is shallow. The more reliable signal lies in the combination of recurrence, aging, leakage after change, and evidence of knowledge reuse.

In order for the sequence to be strict - the first step is to define the business scope of stabilization around a limited set of critical process chains, for example, sales order creation to billing completion, or delivery creation to goods issue and transport confirmation. The second step is to map incident categories to business objects, integration points, batch jobs, and support owners. The third step is to instrument monitoring so that technical alerts can be read alongside business symptoms. At this stage, monitoring practice should be tied to SAP diagnostic transactions already used by support teams, including ST22 for dump analysis, SM37 for background-job failures, and SMQ1/SMQ2 for outbound and inbound queue disruptions. The fourth step is to make root-cause analysis mandatory for repeat incidents above a chosen threshold. The fifth step is to channel resolved cases into reusable documentation and routing rules. The sixth step is to move governance away from pure SLA compliance and toward recurrence reduction, quality of diagnosis, and controlled change introduction.

One of the most underestimated variables in SAP S/4HANA stabilization is documentation design. Documentation often accumulates around project approval and lacks the structure needed for incident diagnosis. Functional specifications, interface descriptions, known error records, monitoring logic, and business ownership maps need a structure that allows support teams to answer three questions quickly: what failed, where in the process chain it failed, and who can change it safely. Once that principle is adopted, documentation becomes an operational tool.

Another issue concerns customization. The literature supports treating customization as an operational and governance risk. In stabilization work, customization becomes dangerous when it obscures accountability, fragments monitoring, or produces process branches that no one actively observes. Every critical deviation from standard process logic requires a visible owner, a traceable signal, and a support path that reaches both functional and technical expertise without delay.

Stable S/4HANA operations emerge when incident management is integrated with process monitoring, problem management, and release governance. Support organizations that isolate these functions will keep solving symptoms. Organizations that connect them can convert operational turbulence into structured system learning.

5. Conclusion

Incident management during SAP S/4HANA stabilization functions as an operational control layer that protects process continuity, reduces recurrence, and links service restoration with organizational learning. The analysis confirms the initial hypothesis: stabilization produces stronger operational results when incident management is embedded in an integrated support model combining business-process monitoring, SAP-specific diagnostic tools, structured ownership, repeat-incident root-cause analysis, and feedback into problem and change governance.

Durable stabilization depends on AMS governance, multi-layer observability, process-level diagnostic visibility, disciplined documentation, and structured feedback from incidents into problem and change management. For application management teams, the implementation sequence starts with critical process scoping and ownership mapping, continues through integrated monitoring and mandatory root-cause analysis for repeat incidents, and matures into knowledge reuse and selective automation.

References

[1] Ahmed, S., Singh, M., Doherty, B., Ramlan, E. I., Harkin, K., Bucholc, M., & Coyle, D. (2023). Knowledge-based intelligent system for IT incident DevOps. In *Proceedings of the 2023 IEEE/ACM International Workshop on Cloud Intelligence & AIOps (AIOps 2023)* (pp. 1–7). IEEE. https://doi.org/10.1109/AIOps59134.2023.00005

[2] Azad, N., & Hyrynsalmi, S. (2023). DevOps critical success factors: A systematic literature review. *Information and Software Technology, 157*, 107150. https://doi.org/10.1016/j.infsof.2023.107150

[3] Berti, A., Park, G., Rafiei, M., et al. (2023). A generic approach to extract object-centric event data from databases supporting SAP ERP. *Journal of Intelligent Information Systems, 61*, 835–857. https://doi.org/10.1007/s10844-023-00799-9

[4] Bradford, M., Bucy, R. A., & Adivar, M. (2024). Implementing ERP with maintenance and support in mind: Exploring the efficacy of application management services for the post-implementation phase. *Journal of Information Systems, 38*(2), 1–27. https://doi.org/10.2308/ISYS-2022-064

[5] Giamattei, L., Guerriero, A., Pietrantuono, R., Russo, S., Malavolta, I., Islam, T., Dînga, M., Koziolek, A., Singh, S., Armbruster, M., Gutierrez-Martinez, J. M., Caro-Alvaro, S., Rodriguez, D., Weber, S., Henss, J., Vogelin, E. F., & Panojo, F. S. (2024). Monitoring tools for DevOps and microservices: A systematic grey literature review. *Journal of Systems and Software, 208*, 111906. https://doi.org/10.1016/j.jss.2023.111906

[6] Holland, R. (2024, March). *SAP S/4HANA migration benchmark report: Executive summary*. SAPinsider. https://sapinsider.org/wp-content/uploads/2025/04/SAPinsider-2024-03-SAP-S4HANA-Migration-Report.pdf

[7] MacLean, D., & Titah, R. (2023). Implementation and impacts of IT service management in the IT function. *International Journal of Information Management, 70*, 102628. https://doi.org/10.1016/j.ijinfomgt.2023.102628

[8] Mahmood, F., Khan, A. Z., Shah, S. A., & Adil, M. (2024). Post-ERP implementation issues and challenges: Exploratory case studies in the context of Saudi Arabia. *Kybernetes, 53*(12), 5749–5774. https://doi.org/10.1108/K-06-2022-0914

[9] Peralta, A., Olivas, J. A., Romero, F. P., & Navarro-Illana, P. (2025). Intelligent incident management leveraging artificial intelligence, knowledge engineering, and mathematical models in enterprise operations. *Mathematics, 13*(7), 1055. https://doi.org/10.3390/math13071055

[10] SAP SE. (2024). *SAP Cloud ALM for Operations - Delta Q2 2024*. https://assets.dm.ux.sap.com/cloud-alm-quarterly-updates/pdfs/sap_cloud_alm_for_operations_delta_q2_2024.pdf

Author Profile

Sudeepta Rana holds an MBA and a Bachelor of Pharmacy degree. He is a Senior Application Consultant with over 17 years of experience in SAP ECC and SAP S/4HANA. His work focuses on OTC transformation, application management, and pharmaceutical supply chain optimization in regulated industries.