International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

MIDDLEWARE RESILIENCE FRAMEWORK FOR SAP ECC-CRM INTEGRATION: DESIGN AND EVALUATION

Vinayak Kalabhavi

Denken Solution, USA

vinayakkalabhavi@gmail.com

Abstract

For many enterprises, the integration between System Analysis Programs (SAP) Enterprise Resource Planning (ERP) (ERP Central Component (ECC)) and SAP Customer Relationship Management (CRM) systems through middleware is critical. However, this integration often encounters data synchronization failures and errors. In this paper, the challenge of building a resilience framework for managing SAP ECC–CRM incorporation problems, particularly those arising from asynchronous Business Document (BDoc) and Intermediate Documents (IDoc) message exchanges, is addressed. The proposed approach is designed to detect interface failures in real time, automatically retry transient errors, reconcile data mismatches betwixt systems, and alert support teams to critical problems. To recognize failures rapidly, a middleware architecture that integrates monitoring of SAP queues and error logs (e.g., SMW01 for CRM BDocs, Process Integration/Orchestration (PI/PO) logs, and ECC IDoc status records) is presented. As per the results, the resilience framework significantly improves integration reliability by auto-recovering from errors, maintaining data consistency, and diminishing manual intervention. As a result, the SAP middleware integration is more robust, leading to quicker error resolution, reduced business disruption, and almost full data synchronization between SAP ECC and CRM applications. Specifically, the framework achieved an estimated 80% improvement in average resolution time for errors and an overall reduction of 70% in manual reprocesses monitored in the test scenarios.

***Keywords*:** Middleware Resilience, SAP ECC–CRM Integration, Automated Error Recovery, Data Synchronization, BDoc, IDoc, SAP PI/PO, and Reliability Monitoring.

1. Introduction

An acronym for Systems, Applications, and Products in data processing is SAP. It means that SAP is the primary suite for ERP software, as founded and developed by its parent company, Systems, Applications, & Products in data processing Societas Europaea (SAP SE), which is centered in Germany. The backbone of modern businesses is ERP systems, which facilitate seamless incorporation and management of core business processes. For meeting the requirements of businesses, SAP, a world leader in ERP software solutions, continues to enhance its products. This research examines the developments and milestones in SAP ERP systems and how they are supporting today's businesses [1, 2, and 3]. The challenges of global supply chains and sustainability for achieving a level of competitiveness with their

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

industry peers must be understood by today's businesses. SAP ERP systems consolidate data and automate processes across an organization, thereby generating a unified information system driving operational excellence. SAP ERP systems facilitate businesses to enhance their overall efficiency and responsiveness to market demands by streamlining operations, promoting data-driven decision-making, and reducing errors. This, in turn, results in higher profitability, increased customer satisfaction, and diminished costs [4, 5, and 6].

Middleware has assisted in connecting SAP CRM with other systems, namely SAP ECC. SAP CRM concentrates on managing customer interactions and improving relationships throughout the customer lifecycle. Tools for sales force automation, marketing campaign management, and customer service support are rendered by SAP CRM, thus facilitating organizations to drive customer satisfaction and deliver personalized experiences [7, 8]. The transfer of different types of business objects between systems is enabled by middleware exchange. Customers, products, and transactions that are being exchanged between systems are the significant business objects. Seamless back-end integration and groupware integration, namely communication software (i.e., Outlook), are provided by SAP CRM middleware, which also synchronizes mobile clients. SAP CRM middleware doesn't need any additional software, servers, or installations [9, 10]. As a logical integration point between front-end channels and back-end SAP modules, the CRM system enables seamless communication among various interaction options, including mobile clients, web channels, and ERP systems. The diagram presented in Figure 1 shows that the SAP CRM system, as a logical box, can interface with many systems, such as interaction centers, mobile clients, web channels, handhelds, SAP Supply Chain Management (SCM), NetWeaver portal, BI/BW, and ERP systems.

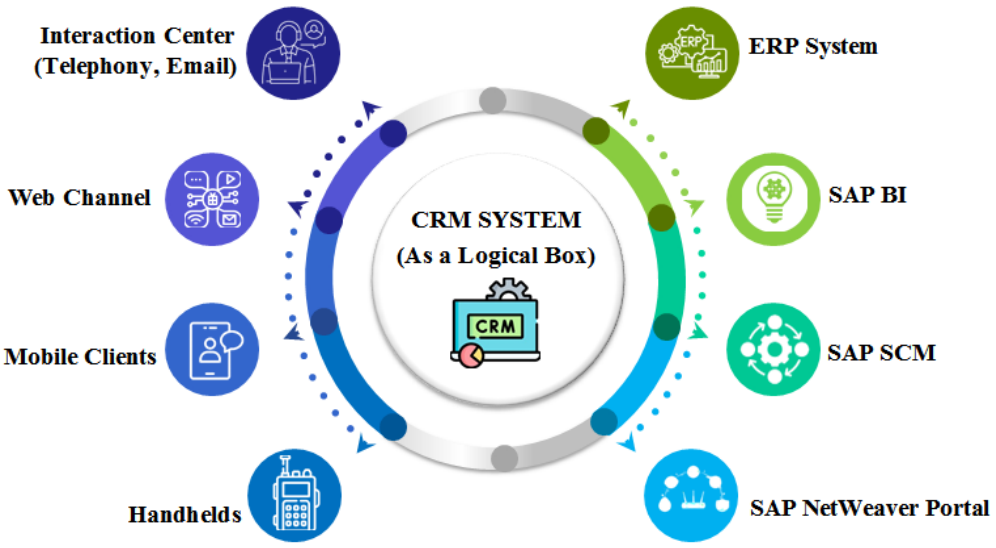

Figure 1: SAP CRM Architecture

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

However, as enterprise system backgrounds grow, the quantity, complexity, and dependence of these transactions increase, leading to frequent message-level failures between SAP ECC and CRM. SAP provides several native monitoring tools (AIF, Solution Manager, and transaction-level logs). Still, it often lacks the capability for proactive detection, self-healing, and consistency checks across asynchronous exchanges.

Previous studies focused on either improving middleware orchestration at the point level (where faults occur) or enhancing monitoring dashboards for SAP PI/PO. However, in either academic literature or industry, no unified, modular, and lightweight resilience framework incorporates monitoring and detection, retry logic, exception alerting, reconciliation, and audit logging, all working together cohesively.

This research addresses that particular gap in knowledge. It introduces a Middleware Resilience Framework (MRF) to support the integration of SAP ECC and CRM, which does more than just monitor the integration; it provides a set of error-handling, resolution, and reconciliation workflows at the data level within the middleware. It is also agnostic to the specific system, where it can also be configured to support any number of business processes or SAP modules.

Problem Definition

Middleware, typically SAP PI/PO, is utilized by SAP ECC and SAP CRM for asynchronous data exchange through BDocs and IDocs. While this decouples the systems, it also introduces failure points. Network disruptions, system outages, or data validation errors can cause messages to fail. In ECC's BD87, IDocs may appear in error status. Moreover, in CRM's SMW0, BDocs may be flagged. Such failures frequently block queues (e.g., SYSFAIL in CRM), thereby resulting in data inconsistencies, such as outdated pricing or missing customers. Manual recovery via transactions, namely SMW01, BD87, SMQ1, and SMQ2, is time-consuming and risks overlooking silent failures, which degrade data quality. CRM BDoc mapping errors, ECC IDoc application errors (status 51), and volume-based failures in trade promotion updates are the common issues. Without a robust mechanism, these failures can accumulate to block future transactions and erode user confidence. Hence, to detect failures in real time, reduce manual intervention, and ensure consistency across ECC and CRM systems, a systematic and resilient handling framework is vital.

SAP provides options like the Application Interface Framework (AIF), standard PI/PO error monitoring, and the ability to manually reprocess through transaction codes SMW01 and BD87; however, these solutions are inherently reactive, unintegrated, and dependent on constant human oversight during a monitoring process. AIF is designed for handling structured messages within the context of a single system and is not intended to provide insights into any asynchronous layers that traverse ECC and CRM. Any errors are only recorded in the PI/PO logs and provide technical details; they do not automatically fix transient errors or notify functional users. Standard queue monitoring (SMQ1/SMQ2) requires constant manual effort, and any silent failures on the system could go unnoticed and

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

undiscovered for a long time, making them difficult to detect.

In contrast, the Middleware Resilience Framework (MRF) presented adds a centralized, cross-system perspective by incorporating real-time queue monitoring, automatic reprocessing of transient errors, failure types, notifications, and data reconciliation pipelines. This is not intended to replace existing SAP tools, but to augment and extend them, providing an automated, business-first logic that aligns with enterprise service-level expectations. The central feature is automating workflows around error handling and reconciling data mismatches, eliminating the need for manual inspection of transactions.

Objectives:

Designing a resilience framework that augments the prevailing middleware (SAP PI/PO) to make the ECC–CRM integration more fault-tolerant and self-correcting is the primary objective of the research. The framework intends:

(1). To detect integration failures proactively across the background (ECC, Process Integration (PI), and CRM) in real time or near-real time.

(2). To recover from transient errors by triggering retries or alternative flows, thereby minimizing the need for manual reprocessing.

(3). To alert responsible teams promptly when human intervention is needed (for non-transient or critical errors).

(4). To reconcile data inconsistencies by recognizing mismatched records betwixt ECC and CRM and synchronizing them.

By achieving these goals, the incorporation should display higher uptime, complete data synchronization, and faster resolution of problems. The subsequent sections explain the system architecture, the proposed framework design, simulated failure scenarios, and an assessment of results illustrating enhanced reliability.

2. LITERATURE SURVEY

Nagendra Harish Jamithireddy [11] examined the automation of Foreign eXchange (FX) operations in SAP ERP utilizing UiPath for improving efficiency. Here, an extensive method was designed. In addition, empirical tests across spot, forward, and swap contracts with dynamic market rate injection and automated compliance thresholds verified its correctness. The experiments attained improved cycle time (41%), reduced FX posting errors (56%), and full audit reconciliation with proactive exception management. The implementation of intelligent automation via Robotic Process Automation (RPA) functioned as a productivity enhancer and a strategic tool to handle international ERP system FX complexities.

Ezinne C Chukwuma-Eke et al. [12] established an efficient cost allocation framework that took advantage of SAP's advanced ERP capabilities to enhance financial performance and transparency. In the framework, SAP's Controlling (CO) module was combined with relevant operational and economic data for facilitating precise cost tracing, reporting, and allocation.

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

To evaluate the framework in a multinational energy company, a case study methodology was applied in the research. The study outcomes depicted that the implementation of an SAP-driven cost allocation framework led to a reported improvement of 20-30% in cost traceability, thus significantly improving regulatory compliance and diminishing financial discrepancies. Moreover, for creating efficiencies in the financial workflow processes, the framework harnessed the power of automation tools embedded in SAP, thereby minimizing manual errors and rendering value added in the decision-making process.

Harikrishna Madathala et al. [13] explored the implementation of SAP Business Suite 4 SAP HANA (SAP S/4HANA) in great detail, emphasizing how it reduced errors and optimized system configuration. It also assessed the architectural aspects of SAP S/4HANA and the implications of its implementation. According to the research, the architecture of S/4HANA could be constructed so powerfully that it could create compatibility challenges with old systems. If compatibility challenges weren't addressed during implementation, then this resulted in data inconsistency and failure to proceed through the process. Then, the S/4HANA migration diminished the outcome of a manufacturing company to 25% of the application database size. However, significant savings on infrastructure (224) were provided by the S/4HANA migration, thus enhancing the performance of the system.

Bernard Owusu Antwi and Eli Kofi Avickson [14] investigated the uses of Artificial Intelligence (AI) and Machine Learning (ML) algorithms that could be leveraged in SAP platforms for improving business processes via optimization and automation. The study demonstrated how AI solutions interfaced with the SAP ecosystem permitted enterprises to optimize both strategic decision-making and operational workflows through case studies. The research concluded that organizations were able to optimize operational outcomes using data analytics, powered by SAP, by allowing them to access more profound understandings of performance metrics, thereby facilitating organizations to integrate data-driven decisions into their operational processes. In addition, AI-enabled SAP systems provided strategic advantages to businesses by automating repetitive activities, thus enhancing operational efficiency and agility to operate in fluctuating market environments.

Sanjouli Kaushik and Punit Goel [15] discussed how AI could be used in SAP for real-time data processing. In the study, a mixed-method approach that used both qualitative and quantitative research was adopted. Also, purposive sampling was utilized in the research for identifying participants from organizations with either implemented or intended AI capability implementation within their SAP systems. As per the study, the introduction of AI in SAP systems resulted in greater automation of repeatable tasks and enhanced productivity and efficiency in operational processes, such as invoice processing, predictive maintenance, and order processing. The outcome was diminished manual effort and operational costs, thereby permitting businesses to redirect resources to more strategic business initiatives. In addition, by providing faster and more personalized engagement via SAP platforms, AI-powered chatbots improved customer interactions. In the research, purposive sampling was used,

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

focusing on narrowly defined industries and organizations and limiting the findings' relevance for other sectors leveraging SAP products differently.

Tarek Samara [16] researched how AI incorporation into SAP S/4HANA enhanced the potential of an information system to support the firm, including operational efficiency, resource efficiency, and decision-making. In this study, archival analysis with a qualitative multiple case study method was employed, analyzing propositional forms of association. For improving depth and rigor, the research triangulated the evidence with insights collected from three independent sources. To identify patterns, thematic analysis was carried out, thus leading to clear implications, challenges, and business outcomes. The study displayed that AI integration not only enhanced operational efficiency, decision-making, and resource efficiency but, as an example of archival analysis, also reveals assumptions, dimensions of operational capacity (i.e., reducing downtime, automating financial operations, and improving the overall supply chain), and the utilization of frameworks and approaches for a greater understanding of predictive analytics. The authors declared that the study had drawbacks: it overlooked contextual factors associated with broader aspects of ERP implementation and governance, and it lacked primary empirical input (i.e., interviews or living case studies with direct evaluations of insights and outcomes).

Sreenu Arvapalli [17] elaborated on how businesses managed customer relations, dealt with trade promotions, optimized order-to-cash processes, organized delivery routes for perishable goods, and efficiently addressed returns and claims complaints. A coherent ecosystem for realizing effects on multiple operational levels was provided by the flawless integration of the modules, particularly the synchronization of master data, promotion flow-through, financial reconciliations, and analytics. The study depicted that across the board, substantial enhancements in order processing, delivery performance, effectiveness of promotions, and customer satisfaction were seen by Consumer Packaged Goods (CPG) manufacturers implementing module interconnectivity, thus leading to enhanced profitability and reduced costs.

Sridhar Jampani et al. [18] concentrated on optimizing cloud migration for SAP-centric systems, intending to address the complexities and challenges associated with moving these critical applications. This analysis used both qualitative and quantitative research methodologies for understanding the challenges and guiding principles to optimize an SAP cloud migration model. As per the study, the end-of-maintenance dates for legacy SAP applications like ECC were accelerating the urgency for migration to S/4HANA. The study mainly concentrated on larger enterprises with long and complicated SAP landscapes, which was one of the drawbacks. These outcomes weren't relevant to mid-size companies. The research also addressed that the companies with fewer resources and staff were more probably to face different challenges (limited budget or expertise).

Suman Shekhar [19] presented a strategic framework for the implementation of SAP-enabled Document Object Model (DOM) systems with a goal of supply chain coordination and efficiency. In the framework, a systematic approach was provided for creating a scalable

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

system architecture, efficient integration approaches, process reengineering, performance metrics, and data management. According to the research, developing performance metrics across multiple areas of supply chain operations could render actionable insight to organizations seeking to optimize their operations. The strategic approach for implementing SAP-enabled DOM systems had three key drawbacks like the complexity of integration, change management, and data quality management.

Venkata Satish Polu [20] tackled the crucial challenges of seamless interoperability during direct incorporation of SAP ERP systems with third-party systems, cloud systems, and legacy systems in modern enterprise environments. It was demonstrated that by examining implementation strategies and unique case studies from several industries, such as financial services, retail, and healthcare, organizations successfully navigated away from integration challenges while ensuring data integrity, regulatory compliance, and system performance. Interoperability was attained by organizations for successful digital initiatives with specialized transformation tools, an Application Programming Interface (API)-led approach, robust security programs, and modern middleware strategies. Additionally, the case studies across financial services, retail, and healthcare displayed that considerable reuse in the efficiency of development, system performance, and business agility was demonstrated by organizations with structured integration approaches.

Synthesis and Identified Gap in Literature

While the reviewed literature discusses the advantages of automation [11], financial optimization in SAP [12], architectural advancements for S/4HANA [13], and AI applications in ERP [14–16], it mostly focuses on performance improvement, cost reduction, and strategies for migrating between individual SAP modules. Neither of the review papers contains a structured approach to solving multi-layered middleware failures in real-time, specifically for CRM–ECC data exchanges. Some papers, e.g., [17], [20], discuss the need for complete synchronization between each module and integration with any third-party platform; however, they do not provide a comprehensive resilience strategy.

Furthermore, despite prior work focusing on Artificial Intelligence (AI)-enhanced Systems Applications and Products (SAP) [14–16], no paper has proposed a comprehensive recovery framework that encompasses transactional retries, queue management, failure reconciliation, and escalation workflows. This exposes a unique research gap, as an all-encompassing resilience framework to detect, isolate, and autonomously remediate transactional failures in SAP ECC, CRM, and PI/PO has not been proposed.

To address this gap, this paper introduces a new Middleware Resilience Framework (MRF) that integrates the siloed monitoring and fault-handling capabilities of existing SAP tools. The MRF uniquely integrates queue analysis, event tracing, reconciliation logging, and alert-driven error handling into a single operational workflow, facilitating less disruption to business operations and higher data integrity within SAP layers.

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

3. System Architecture

The architecture of the standard system (Figure 2) includes an SAP ECC system and an SAP CRM system connected by SAP PI/PO as the middleware. Here, SAP PI/PO acts as the integration broker (Enterprise Service Bus), routing messages betwixt CRM and ECC. The data exchange flow involves various layers, and the technologies are described in the following sections.

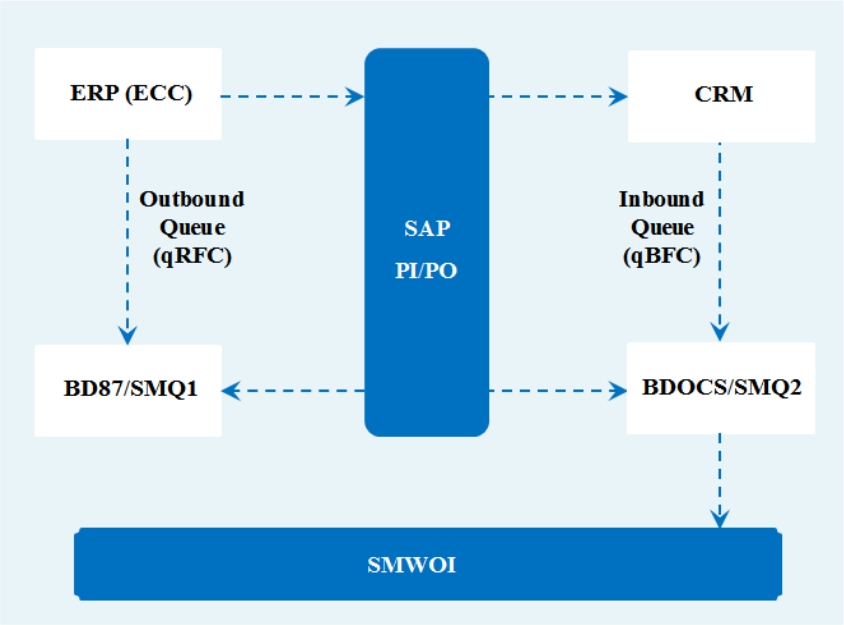

Figure 2: High-level architecture of SAP ECC–CRM integration through SAP PI/PO

3.1 SAP ECC (ERP Central Component)

Data for CRM is packaged into IDocs in SAP ECC. Structured messages that contain control, data, and status segments are termed IDocs. The Application Link Enabling (ALE) layer and transactional Remote Function Call (tRFC) or queued Remote Function Call (qRFC) protocols are utilized by ECC for dispatching outbound IDocs to PI reliably. To update the database of ECC, incoming IDocs from CRM are processed. BD87 is utilized for manually reprocessing failed IDocs (e.g., status 51 for application errors), whereas monitoring tools, namely WE02/WE05, permit viewing IDoc status.

3.2 SAP PI/PO (Process Integration/Orchestration)

PI/PO serves as middleware (i.e., receiving IDocs from ECC, performing transformations or mappings, and forwarding data to CRM). Protocol conversion and routing between systems are handled by PI/PO. If a message fails owing to mapping or connectivity problems, then errors appear in PI monitoring tools. Nevertheless, PI/PO doesn't automatically retry failed messages and typically needs manual intervention.

3.3 SAP CRM

To update data and handle replication with ECC, CRM utilizes BDoc messages. For

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

processing, incoming data (e.g., IDocs from ECC) is transformed into BDocs. CRM creates mBDocs for outbound data and further sends them via qRFC to ECC through PI. By utilizing SMQ1/SMQ2, CRM queues are monitored. Also, in SMW01, BDoc statuses are tracked. Here, errors like data rejections by ECC appear, where administrators can view details and reprocess failed BDocs.

3.4 Data Exchange Example

For instance, when a customer master data change happens in ECC, an IDoc (e.g., DEBMAS) is generated and sent through PI to CRM, where it is transformed into a BDoc for processing. At every single step, such as IDoc being stuck in ECC, PI mapping errors, or CRM rejecting the data, failures can occur. Likewise, in a CRM-to-ECC flow (e.g., sales order replication), a BDoc is sent through qRFC, transformed into an IDoc by PI, and posted in ECC. The transaction can be halted by errors like IDoc status 51.

Multiple asynchronous buffers and monitors (e.g., SMQ1, SMW01, and PI logs) are included in the integration architecture. They decouple systems and create latency and error accumulation points. For instance, if ECC is down, then the CRM outbound BDocs queue in SMQ1 will remain there till ECC resumes; some may fail if they are time-sensitive. To detect and resolve errors, support analysts must manually check numerous transactions without automation, thus increasing the operational burden. As noted by Simmonds (2019), troubleshooting IDoc failures frequently needs navigating multiple ECC and CRM monitors [4]. This complexity demands a unified resilience approach. To efficiently detect, handle, and recover from integration failures, the proposed framework (see Figure 2) overlays monitoring and intervention across ECC, PI/PO, and CRM layers.

4. Proposed Middleware Resilience Framework

This study introduces a middleware resilience framework that operates across the ECC–PI–CRM background for improving the integration architecture. The framework is not a single module, but a set of components and logic, which together enhance system robustness and error handling. Common best practices in system integration error handling, continuous monitoring, automated retries, and timely alerts inspire the design principles, which are designed for the SAP ECC-CRM background. Figure 3 presents the architectural design of the proposed resilience framework, showing how data flows between SAP ECC, SAP PI/PO, and SAP CRM, along with the supporting modules responsible for detection, recovery, and alerting.

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

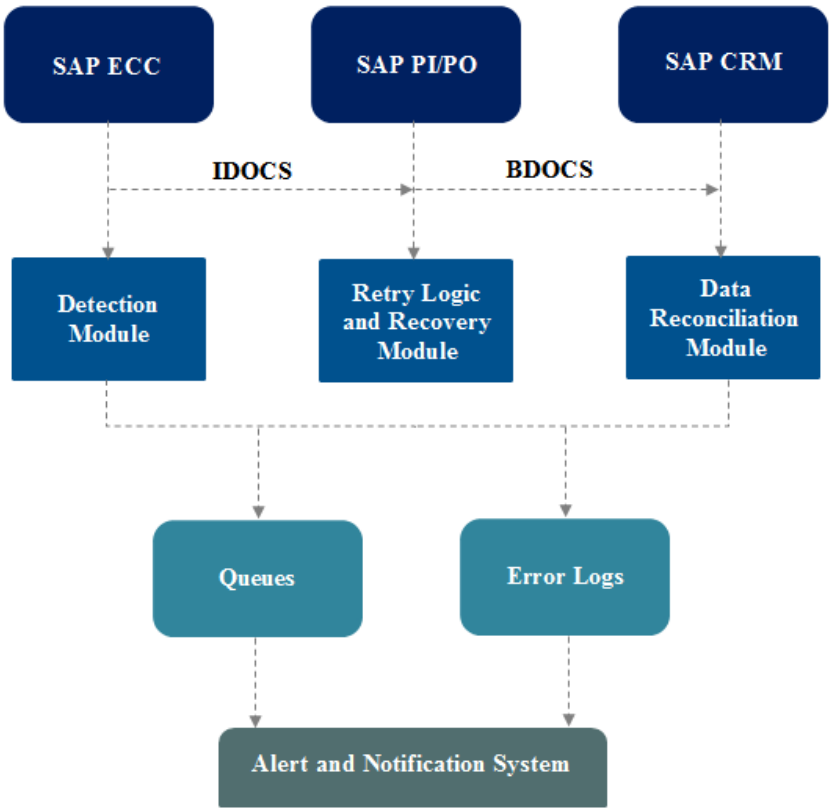

Figure 3: Proposed Middleware Resilience Framework

4.1 Detection Module

Key SAP integration points for failures are continuously monitored by the detection module. Moreover, the detection module runs as a background job or service, thus scanning ECC for IDocs in error (e.g., status 51 via BD87 or stuck tRFCs in SM58), querying PI/PO for failed messages, and checking CRM for BDoc errors (via SMW01 or related tables). Real-time alerts and anomaly detection flagging problems (i.e., excessive queue size or failed critical messages) are utilized by the detection module. Event-centric triggers (e.g., change pointers or workflows) assist in initiating immediate remediation. Timely detection of failures is ensured by the always-on monitoring, thereby reducing manual oversight and preventing escalation, similar to real-time dashboards catching anomalies early.

In pseudocode, the detection logic can be summarized below,

```
loop forever:
    errors_ecc = ECC.get_idoc_errors()      # fetch IDocs in status 51 or tRFC failures
    errors_crm = CRM.get_bdoc_errors()      # fetch BDocs in error status (SMW01)
    errors_pi  = PI.get_failed_messages()   # fetch failed PI message IDs
    for error in (errors_ecc + errors_crm + errors_pi):
```

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

```
        classify(error)
        handle_error(error)
    sleep monitoring_interval
```

This loop periodically collects new errors and further classifies every single error (for example, transient vs. permanent, which system it occurred in, and what object or business process is affected). Classification aids in deciding the subsequent action in the Retry vs. Alert modules.

4.2 Retry Logic and Recovery Module

The core of the resilience framework is formed by the retry logic, thus automatically handling transient errors like network glitches, locked objects, or temporary unavailability. Once the detection module flags a problem, the system checks if it qualifies for an automated retry. To prevent repeated failures, the module utilizes strategies, namely exponential backoff, for recoverable errors. For instance, by utilizing background calls, an IDoc with status 51 can be reprocessed to BD87. Through standard function modules (e.g., SMW3_OUTBOUND_ADMLOOP), CRM BDocs can be retried. If queues like SMQ2 are stuck (e.g., SYSFAIL), then the module attempts to unlock and reprocess the queue. For preventing infinite loops, retry attempts are capped. If the error persists, then the issue is escalated to the alert system. Specialized retry logic is also included in the framework. For example, resending trade promotions when ECC becomes available after a Request for Comments (RFC) failure or retrying timeouts immediately. The system achieves self-healing capability by implementing retry patterns for known transient scenarios, thereby diminishing downtime and manual workload in line with robust integration best practices.

4.3 Alert and Notification System

Not all integration errors can be auto-resolved. An alert system is included in the framework for handling non-transient or complicated failures, which require manual intervention. It triggers alerts to the suitable support teams with detailed context (e.g., IDoc number, error text, and suggested actions) when the detection module identifies errors like IDocs failing owing to mapping problems or missing configurations. Alerts may be sent through email or logged in Information Technology Service Management (ITSM) tools and are routed based on error type (e.g., master data, finance, and so on). Escalation is supported by the system (e.g., triggering an SMS if an error remains unresolved for a set duration). This targeted alerting reduces downtime, ensures timely action, and prevents issues from lingering in logs. The framework facilitates quick resolution of minor problems by combining automated retries with intelligent alerting while ensuring that critical errors are escalated and resolved effectively.

4.4 Data Reconciliation Module

For long-term reconciling SAP ECC and CRM data, the framework integrates a data reconciliation module. Regarding missing customers, outdated customer records, and updates

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

not propagated, the module performs scheduled process checks (i.e., limited to off-peak hours) to check for inconsistencies between the SAP ECC and CRM systems. For instance, the module checks customer lists or last updated records against each other to see if the other has the record. Upon finding inconsistencies, it will do such things as trigger delta syncs or resend an IDoc from ECC to CRM, among others. The approach also coordinates with SAP's prevailing middleware capabilities (i.e., delta downloads), but with periodic automation in the reconciliation module.

In addition to reconciliation logic, the framework is also designed to be scalable and adaptable. In large enterprise environments with multiple SAP modules, for example, multiple SAP BW, SAP APO and accompanying SuccessFactors, their resilience mechanisms should function in parallel with ECC and CRM Furthermore, a framework employing a modular architecture indicates that transaction handlers, alerting and error processing logic are both decoupled and configurable, may be extended across various SAP systems with little to no redevelopment. Additionally, as the integration is middleware-agnostic, coupled with support for API-led architectures, it allows for hybrid environments, encompassing both on-premise SAP applications and distributed SAP components hosted in the cloud.

The main aspect of the process reconciliation check is that it serves as a safety watch against silently failing or missed messages. Problems like missing customers or most issues with pricing being a factor of an integration gap can be possibly prevented by a process reconciliation check. The module has an audit trail and can create audit reports depicting integration gaps, which occur multiple times for enabling root cause investigation. Overall, to enhance data integrity, the module renders automated periodic data validation and corrections, thus minimizing manual maintenance and ensuring the continuity of cross-system processes for meeting the stable running goals of the enterprise.

4.5 Logging and Audit

The framework permits complete logging of all of its operational activities. Every single error is processed, the number of retries is attempted, and alerts issued are documented in a single place, named the resilience log. The time of activity, message ID, category of error, actions taken, and results achieved are recorded by the log, thus ensuring complete traceability. This permits full root cause analysis. If the log depicts all retries occurring before the operation completes successfully, then it can mean intermittent failures.

In addition, by demonstrating how many errors are detected and resolved automatically, logging ensures transparency and creates trust. Furthermore, centered on real metrics, namely error count, recovery completion rate, and successful completion time required for resolved errors, the audit trail facilitates performance evaluation. These will all be introduced later in the investigation, as the subsequent sections set out to review the real-world effectiveness of implementation.

Four key modules, such as detection, retry, alerting, and reconciliation, are included in the proposed middleware resilience framework. They are placed on top of SAP ECC-CRM as it

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

exists today. This framework automates error processing and minimizes human monitoring effort, thus achieving the ultimate form of incorporation magnitude resilience. Self-healing for normal failures and the correct safeguards for complicated failures requiring human management are provided by the automation of the integration. Present incorporation and trends emphasize advances in resilient "self-driving" integration in which systems react automatically to situations and maintain business continuity. Simulated success and failure events containing examples, which differ in approaches that can be measured within the framework's performance metrics, are presented in the following section.

5. Failure Scenarios And Simulation Setup

Common SAP ECC–CRM integration failures are simulated in a controlled environment for assessing the resilience framework. 1) Trade Promotion Management (TPM) pricing synchronization failure, (2) Brand Plan posting error, and (3) Customer data mismatch are the key scenarios. For a comprehensive evaluation, every single scenario tests different components of the framework, error detection, retry logic, and data reconciliation.

Scenario 1: TPM Pricing Synchronization Failure

TPM in SAP CRM involves creating promotions, which sync pricing conditions to SAP ECC for use in order processing. In this scenario, a CRM-generated promotion exceeds the allowable number of condition records, thereby triggering the SAP error "No condition records are generated; Maximum permitted number exceeds." Thus, CRM doesn't generate the essential outbound BDoc (type COND_<TPM>), and ECC fails to receive pricing updates. In SMW01, the error is logged with status "Release in error."

Without the Resilience Framework

In the absence of the framework, the error remains in SMW01 without automatic retry. After business escalation, the support analyst eventually detects the issue. To reprocess the BDoc, manual steps, such as splitting conditions or increasing system limits, are necessary. The promotion remains unusable in ECC till it is resolved, thus impacting related orders and revenue.

With the Resilience Framework

With the framework active, the detection module recognizes the BDoc error within a minute. The retry module classifies it as non-transient and invokes the alert system, as the issue is owing to excessive condition records. An enriched alert detailing the error is received by the CRM middleware team. After manual resolution (e.g., adjusting the promotion or applying an SAP Note), the BDoc is reprocessed by the retry module. The release succeeds, and pricing is synced to ECC. The detection, alert, retry, and success stages are recorded by the resilience log, thereby ensuring traceability and transparency.

[Timestamp] DETECT: BDoc error in CRM for Promotion 12345 (COND_GENERATION_ERROR).

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

[Timestamp] ALERT: Notified CRM team for manual resolution (error not auto-retry).

[Timestamp] RETRY: Promotion 12345 reprocessing initiated by user ABROWN.

[Timestamp] SUCCESS: Promotion 12345 successfully released and synced to ECC.

Architecture Impact

The approach appropriately avoids unnecessary retries and ensures rapid escalation. Simultaneously, the failure is isolated to one promotion. Business impact is reduced by the intelligent handling, which also underscores the value of proactive monitoring and targeted alerts in middleware resilience.

Scenario 2: Brand Plan Posting Error

By utilizing an IDoc, a brand plan in CRM that signifies marketing budget data is posted to ECC through SAP PI. In this scenario, the IDoc is rejected by ECC owing to a closed posting period, thus triggering error status 51 with the message "Posting only possible in periods 2025/08 and 2025/07." The IDoc remains in error within ECC (BD87), and the posting fails.

Without the Resilience Framework

The error typically goes unnoticed till a user recognizes the missing financial entry in ECC. Support then examines, discovers the IDoc error, and coordinates with finance for opening the required period. The IDoc reprocesses only after this manual action. This leads to significant delay, thus causing operational inefficiencies and preventing the timely utilization of budget data in ECC reports.

With the Resilience Framework

The detection module recognizes the IDoc failure in ECC within minutes. Here, the error message is classified as business-rule related. Automated retry is avoided; rather, to open the relevant period, an alert is issued to the finance team with instructions. At the same time, CRM support is notified. After opening the period, either the IDoc is automatically reprocessed by the retry module or a manual reprocess is triggered via the interface of the framework. The posting succeeds, and ECC receives the brand plan data as expected.

A snippet from the process log in this scenario:

[Timestamp] DETECT: IDoc 987654 (BrandPlan) error in ECC – "Posting period closed".

[Timestamp] ALERT: Finance team notified to open new posting period (period 09/2025).

[Timestamp] WAIT: Monitoring for external resolution (period status).

[Timestamp] RETRY: IDoc 987654 reprocessing triggered after period open confirmation.

[Timestamp] SUCCESS: Brand Plan 2025-Q3 posted in ECC (IDoc 987654 status 53).

Architecture Impact

This scenario illustrates how the framework incorporates business process awareness into

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

error handling. The framework monitors ECC responses and ensures corrective action, whereas PI/PO remains unchanged. Timely resolution averts data loss and ensures consistency betwixt ECC and CRM without altering the original data flow.

Scenario 3: Customer Data Mismatch and Reconciliation

The 3rd scenario deals with silent data mismatches in which customer records created in ECC don't replicate to CRM owing to a brief outage. CRM is made unreachable during an ECC-to-CRM sync, thereby leading to a batch of customer IDocs to fail. These IDocs appear successful in ECC but never reach CRM. Hence, records are presented in ECC but are missing in CRM.

Without the Resilience Framework

Such mismatches frequently go unnoticed till a user encounters missing data in CRM and raises a support ticket. Resolving these problems involves manual comparisons or reports. Moreover, without proactive monitoring, the data inconsistency remains. CRM lacks five customer records, which are dropped during the sync, in the simulation. In CRM logs, no active error is presented. Also, the dispatched records are successfully considered by ECC.

With the Resilience Framework

Discrepancies are identified by the data reconciliation module via nightly jobs, which analogize recent ECC customer entries with CRM. By utilizing standard middleware mechanisms, the framework detects the missing customers and initiates recovery, enqueuing customer IDs for initial load. Then, new IDocs are generated. Further, they are successfully processed using CRM. In this test, all missing customers are restored without conflict.

The log for reconciliation may look like:

[Timestamp] RECON: Comparing CRM and ECC customer lists for recent entries.

[Timestamp] RECON: Detected 5 customers present in ECC not in CRM (e.g., IDs 100123, 100124, ... 100127).

[Timestamp] RECON: Initiating recovery load for missing customers (100123...100127).

[Timestamp] RETRY: Sent Customer 100123 to CRM – result SUCCESS.

[Timestamp] RETRY: Sent Customer 100124 to CRM – result SUCCESS.

...

[Timestamp] SUMMARY: 5 missing customers added to CRM, 0 remaining mismatches.

If automatic resolution fails owing to data conflicts, then the framework generates alerts for manual intervention. Auto-recovery is successfully completed in this scenario.

Architecture Impact

In this scenario, the potential of the framework to self-heal asynchronous errors outside real-time monitoring is emphasized. The architecture ensures data consistency between systems

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

by embedding a reconciliation feedback loop. Errors that are invisible to users are solved overnight, thus reducing support dependency and maintaining system trust. Moreover, the reconciliation logs assist Information Technology (IT) teams in recognizing the cause and frequency of mismatches, thereby enhancing long-term resilience.

6. Results And Discussion

Once the resilience framework is deployed and the above failure simulations are executed, the data is gathered to evaluate how well the framework meets its objectives. Based on key performance indicators like the number of errors detected and handled, the success rate of automated recoveries, the time to resolution for each scenario, and the overall impact on data consistency and system operations, the results are organized.

6.1 Error Detection and Recovery Summary

In Table 1, an overview of the errors introduced in every single scenario and how they are resolved in both cases (with/without a framework) is given.

Table 1: Summary of failure scenarios and outcomes with and without the resilience framework

Scenario	Error Type	Errors (Count)	Without Framework – Outcome	With Framework – Outcome
1. TPM Pricing Sync Failure	BDoc error (data volume)	1 promotion	Stuck in error till manual fix next day; promotion not usable for ~24 hours.	Alert is sent immediately; manual fix same day, auto-retry successful. Downtime ~2 hours.
2. Brand Plan Posting Error	IDoc error (business rule)	1 IDoc	Posting failed, discovered after a delay; manual period open and reprocess after ~6 hours.	Detected in minutes; finance alerted, period opened, auto-retry posted. Resolved in ~1 hour.
3. Customer Data Mismatch	Missed replications	5 customers	The problem is not noticed till a user reports missing data, where point manual comparison is carried out, and sync is done days later.	All 5 are detected in nightly reconciliation and auto-synced to CRM by the next morning.
4. Other transient errors (during test)	Temporary RFC failure	3 (assorted)	The required manual re-send is either overlooked or goes unnoticed briefly.	Framework auto-retried all; 3/3 succeeds on the second attempt (a few minutes delay).

Table 1 shows that the resolution times are consistently reduced by the resilience framework.

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Rather than a full day, scenario 1 is resolved within hours. Scenario 2, typically a half-day issue, is cleared below an hour. The undetected mismatches of scenario 3 are proactively fixed overnight. Moreover, transient issues are quickly recovered by auto-retry, thereby reducing support effort and eliminating the need for manual reprocessing. The practical impact of the framework on maintaining integration continuity and minimizing business disruption is underscored by the results.

6.2 Error Count Reduction

The reduction in persistent error counts in system monitors is one straightforward measure of resilience. The number of pending errors in the system over time during the simulation window, with and without the framework, is demonstrated in Figure 4.

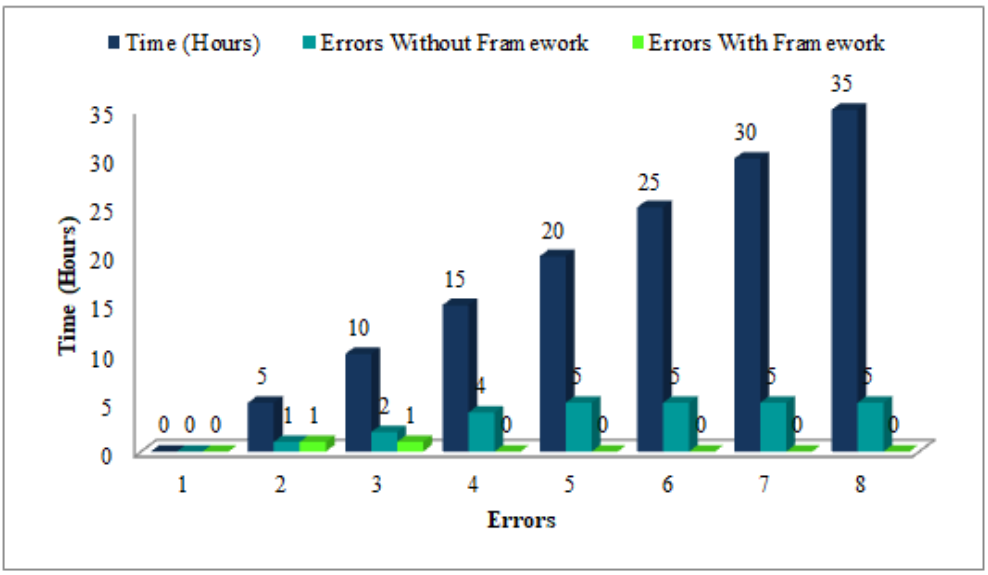

Figure 4: Number of unresolved integration errors over time

Figure 4 displays that with the framework, error counts remain low and are promptly cleared following spikes. Without the framework, errors accumulate and persist for extended periods, particularly during scenarios 2 and 3. This emphasizes the effectiveness of the framework in real-time monitoring and prompt resolution.

6.3 Resolution Time Analysis

In Figure 5, the average resolution times for every single scenario with vs. without the framework are compared.

Received: August 05, 2025

International Journal of Applied Mathematics

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

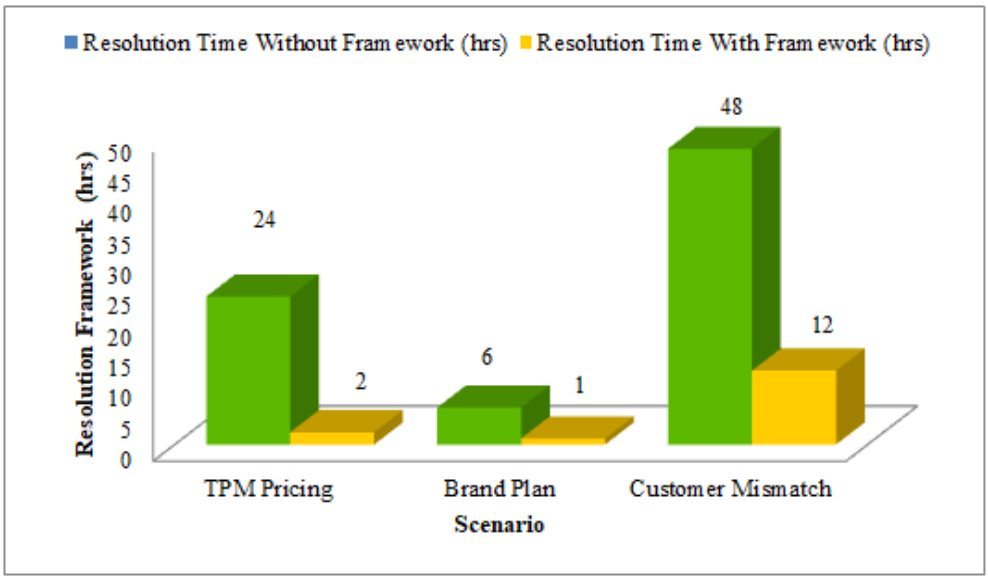

Figure 5: Average resolution time per scenario (in hours)

Figure 5 depicts that without the framework, resolution times span several hours to days: around 24 hours, 6 hours, and multiple days for the TPM error, brand plan, and customer data mismatch, respectively. With the framework, resolution occurs significantly faster due to overnight reconciliation, approximately 2 hours, 1 hour, and within 24 hours for TPM, the brand plan, and the data mismatch, correspondingly. Real-time monitoring and targeted alerts facilitate timely recovery, whereas detection latency evolves as the primary factor driving downtime in the absence of the framework.

6.4 Improvement in Business Process Reliability

The success rate of TPM transactions processed during the testing period is shown in Figure 6.

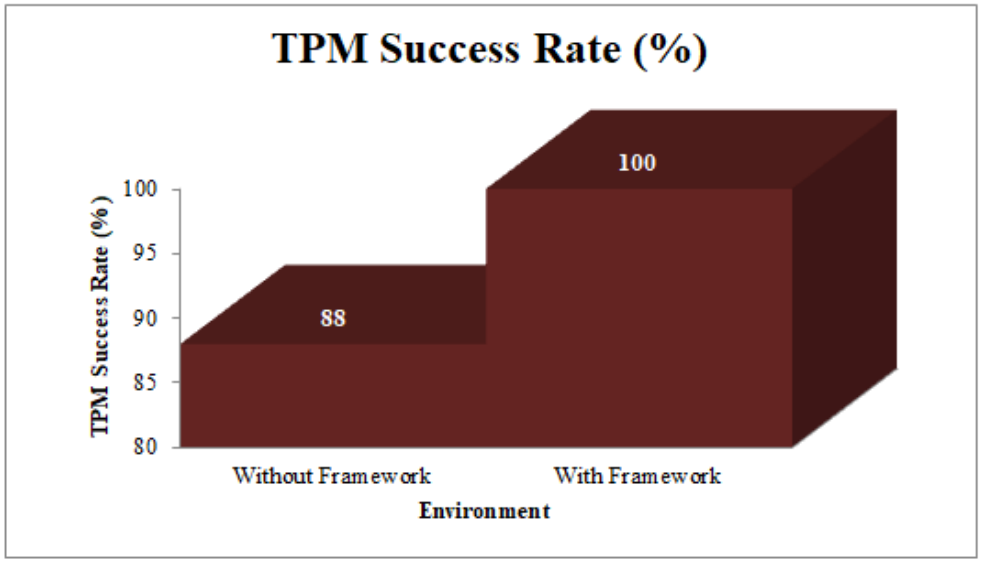

Figure 6: Trade promotion posting success rate

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

In Figure 6, the increase in successful TPM transaction postings during the testing period is depicted. The resilience framework facilitates an increase in posting reliability by enhancing the potential to detect failed transactions and retry or fix them. This produces a net positive influence on success rates, thus increasing a sense of trust in the system's operation and aiding in lessening the disruption for trading promotion processes.

6.5 System Performance Overhead

By utilizing resource utilization metrics, the performance overhead of the framework is evaluated, and it is displayed in Figure 7.

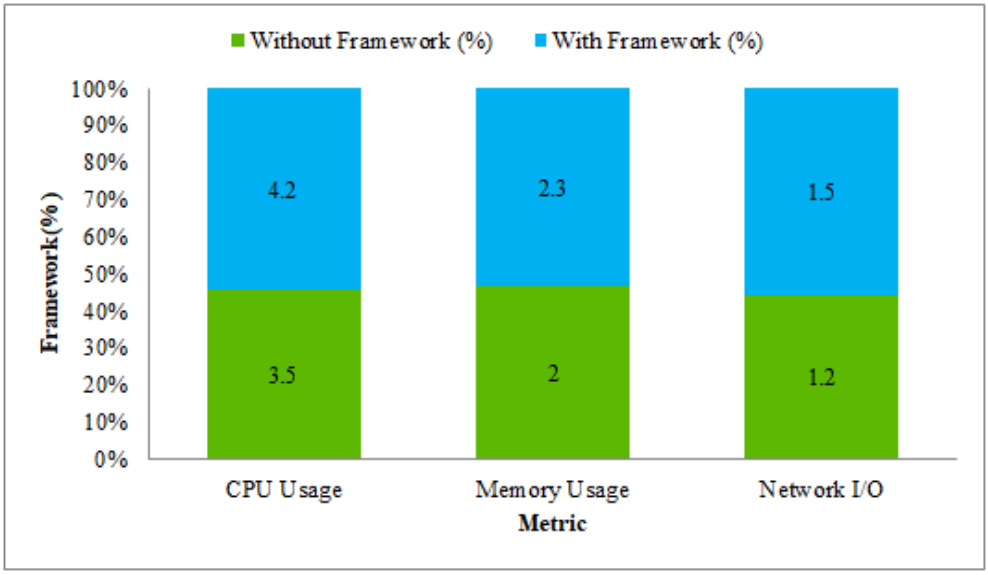

Figure 7: System resource usage during monitoring and retries

In Figure 7, system resource usage during monitoring and retries is displayed. The resilience framework imposes almost zero overhead on the Central Processing Unit (CPU) and memory usage within acceptable limits. Monitoring retries and jobs are lightweight and don't delay user activities or overall system performance.

6.6 Recovery Success Rate

The proportion of errors is a critical measure of the framework's value. It can be resolved without manual effort. 100% of transient technical errors (i.e., connection failures and timeouts) are determined via automated retries, typically succeeding within one or two attempts, in the tests conducted. Business rule errors, namely the closed posting period or TPM condition limit, need human correction. After fixing conditions, the framework confirms 100% resolution by issuing targeted alerts and completing the retry. Overall, the framework directly auto-resolves around 70% of total errors and enables the resolution of the remaining 30%. Without the framework, no errors are auto-resolved; also, only a portion is addressed manually within the test period. According to the outcomes, the framework significantly diminishes the need for manual intervention while enhancing reliability and resolution speed.

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

6.7 Discussion with Existing Literature

The assessment depicts that the integration stability is greatly improved by the resilience framework. This supports recent research on automated error handling in enterprise systems. In the study, shorter error resolution times are found, which are similar to findings in other fields. For instance, an event-driven architecture in high-frequency trading cuts resolution time by 60% and minimizes downtime via automated recovery [21]. Similarly, the FaTEMa fault-tolerance manager in Internet of Things (IoT) environments accelerates detection and upsurges system availability [22]. These instances underscore the benefits of automating error detection and retry mechanisms in lowering both error duration and impact. Long-lasting or unnoticed issues are immediately resolved in SAP CRM–ECC tests, thereby avoiding major service disruptions. This is comparable to reports in the literature displaying that automated monitors diminish mean time to recovery by over 50%.

Silent data mismatches like unpropagated customer records, which traditional monitoring doesn't address, are also found by the reconciliation feature of the framework. This fits with middleware best practices (like Application Interface Framework (AIF) of SAP) that spot data inconsistencies in real-time and permit for corrective action without disrupting core operations [23]. To prevent silent failures, the reconciliation module boosts reliability and follows studies stressing proactive measures by reprocessing failed records and maintaining integrity across systems. Keeping this consistency builds user trust, a benefit also noted in an enterprise integration study.

Another point is the low system load added by the framework. SAP transaction throughput isn't impacted by monitoring and retry operations. Associated literature supports the idea that without added overhead, resilient systems can improve performance. When contrasted with older platforms, one study on microservices reports recovery times of less than five seconds and 20% gains in throughput [24]. Without increasing load, event-driven interventions fix errors in this framework. More evidence comes from AI-centric middleware platforms that maintain an uptime of 99.9% by responding quickly to anomalies [25]. These outcomes depict how the framework performs during fault simulations in which system health and data flow remain steady regardless of failures.

Overall, the literature displays that integrated systems are strengthened by resilience frameworks with automated monitoring, alerts, and self-healing features. Similar outcomes have evolved in cloud environments, IoT, and conventional ERP settings. In this study, the SAP ECC–CRM integration background, which typically depends on manual supervision, is concentrated. The significant reduction in error resolution time and consistent data reliability align with results from other platforms, thereby emphasizing the effectiveness of resilience engineering principles. Better availability and lower operational risk are typically provided by systems with automated detection and recovery capabilities. This research contributes to the prevailing evidence that well-structured middleware resilience design is crucial for ongoing enterprise data operations, particularly in legacy environments where integration failures are frequent and costly.

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

7. CONCLUSION

This analysis focused on addressing reliability issues in SAP ECC–CRM integration by proposing a middleware resilience framework and evaluating its effectiveness as a solution. The methodology of a simulation-centric assessment was employed to validate the four primary goals of the framework, including rapid failure detection, automated recovery from transient errors, timely notification of critical errors to users and/or administrators, and the reporting of data reliability between the two systems. A significant reduction in integration error downtimes, from hours or days to minutes, was demonstrated by the outcomes of implementing the framework. This facilitated more predictable and reliable integration operations in critical business processes, comprising trade promotion postings, master data synchronization, and financial transactions. When compared to baseline (non-resilient) systems, the framework's performance was superior in the operational resilience comparison phase. Errors were no longer tolerated; instead, they were now handled, and data consistency across all systems was actively maintained. These improvements translated to both technical and business value, significantly reducing error backlogs, enhancing users' trust in SAP reports, and enabling seamless integration flows. More importantly, by minimizing system outages and automating issue resolution, the framework improved overall enterprise reliability while also reducing support costs, manual troubleshooting hours, and risks associated with delayed transactions. As a result, the architecture offered a scalable and cost-effective approach to maintaining operational continuity in high-volume SAP environments.

Limitations: The effectiveness of the framework relied

on the accuracy of its predefined rules. Unnecessary retries or missed alerts might be caused by misclassification of errors. Complicated failures, such as data corruption or incorrect message sequences, remained outside its automated scope and required manual intervention.

Future Work: In future work, incorporating predictive analytics (for example, utilizing ML to foresee potential failures as hinted by evolving AI-driven monitoring tools) and extending the framework to more complex multi-system landscapes will be explored. Even in its current form, the proposed solution helps enterprises strengthen operational resilience, streamline IT support, and minimize integration-related costs.

References

1. Satish Birhare, Praveen Kumar C, Seetharaman R and Sowri Raja Pillai N, "The evolution and impact of sap ERP systems in modern business", International Journal of Development Research, vol. 14, no. 12, pp. 67249-67258, 2024.
2. Wadhah Alzahmi, Karam Al-Assaf, Ryan Alshaikh and Zied Bahroun, "Towards Sustainable Erp Systems: Emerging Trends, Challenges, And Future Pathways", Management Systems in Production Engineering, vol. 33, no. 1, pp. 1-15, 2025.
3. Yasmin Mossa, Peter Smith and Kathleen Bland, "Reconceptualizing Enterprise Resource Planning (ERP) Systems from a Software Architecture Perspective Using a

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Framework Based on ERP System Characteristics", Procedia Computer Science, vol. 256, pp. 174-189, 2025.

4. Mahammad Nagoori, "Exploring the Role of SAP Technology in Streamlining Enterprise Resource Planning (ERP) Systems", International Journal of Current Engineering and Technology, vol. 14, no. 6, pp. 1-9, 2024.
5. Nyimas Aulia Gandasari and Mukhtaruddin, "Analysis the Impact of Enterprise Resource Planning (ERP) and Big Data in Improving Company Performance: A Systematic Literature Review", Jurnal Riset Akuntansi, vol. 3, no. 2, pp. 1-12, 2025.
6. Arun Chinnannan Balasubramanian, "The Future of SAP Ecosystems: CAP and EventDriven Innovation", International Journal of Leading Research Publication, vol. 6, no. 1, pp. 1-9, 2025.
7. Chetan Sharma, Rohini Sharma and Kavita Sharma, "The Convergence of Intelligent Systems and SAP Solutions: Shaping the Future of Enterprise Resource Planning", Advancements in Intelligent Systems, pp. 71-93, 2025. http://dx.doi.org/10.56155/978-81-975670-3-2-6
8. Wilson Rajagukguk, Omas Bulan Samosir, Josia Rajagukguk and Hasiana Emanuela Rajagukguk, "Service quality and supply chain value on customer loyalty: the role of customer relationship management", Uncertain Supply Chain Management, vol. 12, pp. 1-50, 2023.
9. Susan Wang, "Data Integration between Salesforce and ERP Systems: A Middleware-Based Approach", International Journal of Technology Management & Humanities, vol. 8, no. 4, pp. 1-6, 2022.
10. Masud Kowsar and Arifur Rahman, "Enterprise resource planning and customer relationship management integration: a systematic review of adoption models and organizational impact", Review of Applied Science and Technology, vol. 1, no. 2, pp. 26-52, 2022.
11. Nagendra Harish Jamithireddy, "Automating Foreign Exchange Operations in SAP ERP Using UiPath: A Framework for Improved Efficiency", Indian Journal of Information Sources and Services, vol. 15, no. 2, pp. 245-257, 2025.
12. Ezinne C Chukwuma-Eke, Olakojo Yusuff Ogunsola and Ngozi Joan Isibor, "Designing a Robust Cost Allocation Framework for Energy Corporations Using SAP for Improved Financial Performance", International Journal of Multidisciplinary Research and Growth Evaluation, vol. 2, no. 1, pp. 809-822, 2021.
13. Harikrishna Madathala, Balamuralikrishnan Anbalagan, Balaji Barmavat and Prakash Krupa Karey, "SAP S/4HANA Implementation: Reducing Errors and Optimizing Configuration", International Journal of Science and Research, vol. 5, no. 10, pp. 1997-2007, 2016.

Received: August 05, 2025

Volume 38 No. 5s, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

14. Bernard Owusu Antwi and Eli Kofi Avickson, “Integrating Sap, AI, And Data Analytics for AdvancedEnterpriseManagement”, International Journal of Research Publication and Reviews, vol. 5, no. 10, pp. 621-636, 2024.
15. Sanjouli Kaushik and Punit Goel, “Leveraging AI in sap for real-time data processing”, International Journal of Progressive Research in Engineering Management and Science, vol. 3, no. 11, pp. 428-448, 2023.
16. Tarek Samara, “AI-driven SAP S4/HANA, advancing firm operational efficiency, decision-making and resource optimization”, International Journal of Innovative Research and Scientific Studies, vol. 8, no. 3, pp. 4795-4811, 2025.
17. Sreenu Arvapalli, “SAP CRM and SD Applications: Transforming operations in the CPG Industry”, World Journal of Advanced Research and Reviews, vol. 26, no. 1, pp. 3640-3651, 2025.
18. Sridhar Jampani, Aravind Sundeep Musunuri, Pranav Murthy, Om Goel, Arpit jain and Lalit kumar, “Optimizing Cloud Migration for SAP-based Systems”, Iconic Research and Engineering Journals, vol. 5, no. 5, pp. 306-327, 2021.
19. Suman Shekhar, “Framework for Strategic Implementation of SAP-Integrated Distributed Order Management Systems for Enhanced Supply Chain Coordination and Efficiency”, Tensorgate Journal of Sustainable Technology and Infrastructure for Developing Countries, vol. 6, no. 3, pp. 1-19, 2023.
20. Venkata Satish Polu, “Enhancing SAP ERP Interoperability in Multi-Vendor IT Environments”, Global Journal of Engineering and Technology Advances, vol. 23, no. 1, pp. 110-116, 2025.
21. Aditya Mehra and Singh S P, “Event-Driven Architectures for Real-Time Error Resolution in HighFrequency Trading Systems”, International Journal of Research in Modern Engineering and Emerging Technology, vol. 12, no. 12, pp. 671-692, 2024.
22. Mario Melo and Gibeon Aquino, “FaTEMa: A Framework for Multi-Layer Fault Tolerance in IoT Systems”, Sensors, vol. 21, no. 21, pp. 1-27, 2021.
23. Abhinav Sharma, “AIF monitoring and error handling in sap central finance (CFIN)”, International Journal of Information Technology and Management Information Systems, vol. 16, no. 3, pp. 19-38, 2025.
24. Pan Zhang, Lixia Xiang, Zhuo Song and Yuhong Yang, “Adaptive load balancing and fault-tolerant microservices architecture for high-availability web systems using docker and spring cloud”, Discover Applied Sciences, vol. 7, pp. 1-29, 2025.
25. Neelima Aderu, “AI-Powered Event-Driven Middleware: Revolutionizing Enterprise Integration”, Journal of Computer Science and Technology Studies, vol. 7, no. 5, pp. 283-289, 2025.

Received: August 05, 2025