**ISSN:** 2583-9888
**DOI:** doi.org/10.51219/JAIMLD/purnachander-rao-nadipelli/692

URF PUBLISHERS
connect with research world

Journal of Artificial Intelligence, Machine Learning and Data Science

https://urfpublishers.com/journal/artificial-intelligence

Vol: 6 & Iss: 1

*Research Article*

# Clean Core Architecture: The Foundation for Sustainable SAP S/4HANA Transformations

**Purnachander Rao Nadipelli***

IT Technical Manager, ERP Transformation, SAP S/4HANA, BTP - Corning Inc. USA

**Citation:** Nadipelli PR. Clean Core Architecture: The Foundation for Sustainable SAP S/4HANA Transformations. *J Artif Intell Mach Learn & Data Sci 2023* 6(1), 3517-3524. DOI: doi.org/10.51219/JAIMLD/purnachander-rao-nadipelli/692

**Received:** 02 March, 2023; **Accepted:** 18 March, 2023; **Published:** 20 March, 2023

***Corresponding author:** Purnachander Rao Nadipelli, IT Technical Manager, ERP Transformation, SAP S/4HANA, BTP - Corning Inc., USA



## ABSTRACT

The accumulation of custom code, unreleased interfaces, and tightly coupled extensions continues to weaken the maintainability, scalability, and upgrade readiness of SAP S/4HANA environments. This study examines Clean Core Architecture as a foundation for sustainable SAP S/4HANA transformation. Using a conceptual and architecture-based analytical approach, it evaluates Clean Core principles, ABAP Cloud, released APIs, Embedded Steampunk, side-by-side extensions, and SAP Business Technology Platform extensibility. The paper argues that sustainable transformation requires systematic decisions regarding which capabilities should remain within the digital core, be refactored, retired, replaced, or externalized. It proposes a governance framework combining architectural standards, customization assessment, extension approval processes, accountability structures, compliance monitoring, and technical debt controls. The analysis indicates that effective Clean Core adoption can reduce maintenance complexity, improve upgrade compatibility, strengthen system resilience, and support continuous innovation. However, these outcomes depend on disciplined migration planning, business–IT alignment, and long-term governance. Clean Core should therefore be understood not merely as a technical architecture, but as an enterprise operating model for sustaining SAP S/4HANA transformation value.

**Keywords:** Clean Core Architecture, SAP S/4HANA, SAP BTP, ABAP Cloud, technical debt, ERP governance

## 1. Introduction

Enterprise resource planning systems frequently evolve through years of configuration, custom code, interface development, and process-specific modifications. Although such adaptations may initially improve organizational fit, they also create tightly coupled architectures that are difficult to maintain, test, and upgrade. ERP customization therefore represents a strategic trade-off rather than a purely technical choice. Brehm et al. (2001) showed that organizations can select from a broad spectrum of tailoring options, each carrying different implications for control, flexibility, and long-term system complexity. Subsequent studies similarly indicate that customization decisions are shaped by organizational misfit, business requirements, and implementation realities, but often produce dependencies that persist beyond the original transformation context (Haines, 2009; Hustad et al., 2016; Rothenberger & Srite, 2009).

This challenge becomes particularly significant during SAP S/4HANA transformation. Legacy customizations may rely on obsolete objects, undocumented interfaces, duplicated business logic, or modifications embedded directly within the ERP core. As a result, migration is not simply a technical conversion exercise; it requires organizations to determine which custom developments remain valuable, which should be redesigned, and which should be eliminated. Parthasarathy and Daneva (2016) emphasize the importance of prioritizing requirements when estimating ERP customization, while Feldman et al. (2016) demonstrate that upgrade processes are strongly affected by the condition and complexity of existing enterprise systems. In this context, extensive customization can delay transformation, increase testing demands, raise implementation risk, and weaken future upgrade compatibility.

The problem can also be understood through the concept of technical debt. Technical debt emerges when short-term design decisions create future maintenance obligations and architectural constraints (Kruchten et al., 2012). Its accumulation is associated with reduced maintainability, higher remediation costs, and increased difficulty in managing system evolution (Li et al., 2015; Rios et al., 2018). Within SAP environments, unreleased interfaces, direct modifications, redundant code, and tightly coupled extensions may provide immediate functional benefits while creating long-term liabilities. Clean Core Architecture responds to this problem by promoting standardization, controlled extensibility, stable interfaces, and separation between the transactional core and organization-specific innovation.

The significance of this study lies in positioning Clean Core not merely as a software design preference, but as a governance mechanism for sustainable enterprise transformation. Enterprise architecture research has long argued that standardized platforms and clearly defined decision rights are essential for business execution and long-term adaptability (Ross et al., 2006; Weill & Ross, 2004). This perspective is increasingly important as digital transformation requires organizations to balance operational stability with continuous innovation (Verhoef et al., 2021; Vial, 2019). Legacy systems and fragmented data architectures can further constrain this balance by limiting scalability and slowing organizational responsiveness (Cao & Iansiti, 2022).

Accordingly, this paper examines Clean Core Architecture as a foundation for sustainable SAP S/4HANA transformation. It critically evaluates the architectural logic of minimizing core modifications, using stable extension mechanisms, and governing customization decisions across the system lifecycle. The study further aims to analyze how legacy customizations can be assessed and migrated, how Clean Core practices can reduce technical debt and improve upgrade compatibility, and how an integrated governance framework can support long-term ERP sustainability.

## 2. Literature Review and Conceptual Foundation

### 2.1 Evolution of SAP ERP Customization

ERP customization developed to address gaps between standardized software and organization-specific processes. Tailoring ranges from configuration to source-code modification, with each option creating different levels of dependency, risk, and lifecycle cost (Brehm et al., 2001). Although customization may improve short-term fit, excessive adaptation can preserve inefficient processes and embed structural misalignment. Enterprise-system fit is multidimensional; process redesign and governance reform may therefore be more effective than modification alone (Strong & Volkoff, 2010). Customization is also shaped by implementation pressure, organizational politics, and perceived differentiation, causing ERP landscapes to evolve through fragmented choices rather than coherent architecture (Haines, 2009; Rothenberger & Srite, 2009). These dependencies become problematic during upgrades, where they increase testing and remediation effort (Feldman et al., 2016; Hustad et al., 2016). Cloud ERP has consequently accelerated the movement from unrestricted modification toward controlled extensibility, although this introduces new governance and dependency trade-offs (Abd Elmonem et al., 2016; Schneider & Sunyaev, 2016).

### 2.2 Clean Core Principles

Clean Core Architecture seeks to preserve SAP S/4HANA stability by keeping the transactional core close to standard while implementing differentiated capabilities through governed, upgrade-stable mechanisms. Standardization is the default, and deviations should be justified by strategic, regulatory, or operational value (Parthasarathy & Daneva, 2016). Side-by-side extensions on SAP BTP separate innovation from the core but may relocate complexity into integration and operational coordination when governance is weak (Soldani et al., 2018). Released APIs and supported extension points reduce coupling to unstable internal objects and establish clearer architectural boundaries (Ross et al., 2006). ABAP Cloud and Embedded Steampunk reinforce this discipline by restricting development to approved objects and services. Their effectiveness depends on architectural review, interface control, extension inventories, and assigned decision rights (Weill & Ross, 2004).

### 2.3 Technical Debt in ERP Systems

Technical debt represents future costs created by short-term decisions that weaken software quality (Kruchten et al., 2012). In ERP environments, it extends beyond code to architecture, testing, documentation, and organizational dependency (Tom et al., 2013). Custom developments may remain embedded in critical workflows after their value has declined, making SAP S/4HANA migration both a technical and business challenge. Effective debt management requires identification, measurement, prioritization, repayment, and prevention (Li et al., 2015). Clean Core can contain debt by limiting unsupported dependencies and promoting stable interfaces, but externalization does not eliminate risk. Poorly governed SAP BTP extensions may create integration, security, and maintenance liabilities. Clean Core should therefore be treated as a continuous debt-governance strategy rather than a guarantee of debt-free architecture (Rios et al., 2018).

### 2.4 Research Gaps

The literature shows that ERP customization can improve organizational fit while increasing complexity, dependency, technical debt, and upgrade effort. However, relevant insights remain fragmented across customization, cloud ERP, technical-debt, enterprise architecture, and governance research. Existing studies also largely predate SAP S/4HANA extensibility mechanisms such as ABAP Cloud, Embedded Steampunk, released APIs, and SAP BTP side-by-side development. Consequently, they provide limited guidance for selecting among standard functionality, embedded extensions, external applications, and legacy-code retention. Migration research similarly lacks an integrated basis for deciding whether custom developments should be retained, retired, replaced, refactored, or externalized, particularly when business value, architectural coupling, interface stability, and upgrade risk must be considered (Parthasarathy & Daneva, 2016). Governance research emphasizes decision rights and accountability but does not translate them into a complete Clean Core model (Ross et al., 2006; Weill & Ross, 2004). This study addresses these gaps through an integrated governance framework for sustainable SAP S/4HANA transformation.

**Table 1.** Comparison of Key Literature Relevant to Clean Core Architecture.

| Study | Primary focus | Key contribution to the present study | Limitation |
|---|---|---|---|
| Brehm et al. (2001) | ERP tailoring alternatives | Establishes that customization choices produce different levels of complexity and dependency | Predates cloud ERP and SAP S/4HANA extensibility models |
| Luo and Strong (2004) | Evaluation of ERP implementation choices | Demonstrates the need to balance organizational fit, cost, and flexibility | Focuses primarily on implementation rather than lifecycle sustainability |
| Haines (2009) | Organizational influences on ERP customization | Shows that customization is shaped by contextual and political factors | Provides limited guidance for governing customizations after implementation |
| Rothenberger and Srite (2009) | Customization decisions in ERP projects | Links customization to business requirements and perceived differentiation | Does not address cloud-compatible extension strategies |
| Kruchten et al. (2012) | Technical-debt theory | Provides the conceptual basis for analysing future costs of customization | Not specific to ERP environments |
| Li et al. (2015) | Technical-debt management | Identifies debt-management activities such as measurement, prioritization, and repayment | Offers limited treatment of enterprise governance and business-process dependency |
| Hustad et al. (2016) | ERP misfit and customization | Demonstrates how customization can evolve into long-term organizational and technical dependency | Based on customization experience rather than Clean Core transformation |
| Feldman et al. (2016) | Enterprise-system upgrades | Highlights the organizational and technical complexity of upgrade processes | Does not propose an architecture for reducing customization-related upgrade effort |
| Soldani et al. (2018) | Microservices benefits and challenges | Supports critical evaluation of side-by-side and modular extensions | Based largely on grey literature and not focused on SAP environments |
| Cao and Iansiti (2022) | Legacy systems and data architecture | Connects legacy complexity with constraints on digital transformation | Does not examine SAP-specific extensibility or governance mechanisms |

**Figure 1.** Conceptual Framework for Sustainable Clean Core Transformation.

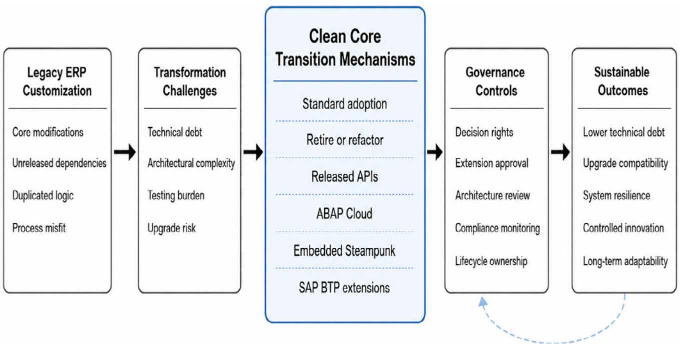


Figure 1 illustrates how legacy SAP ERP customization generates architectural complexity, technical debt, testing burdens, and upgrade risks. Clean Core transition mechanisms, supported by continuous governance controls, convert these constraints into sustainable SAP S/4HANA outcomes, including improved upgrade compatibility, resilience, controlled innovation, and long-term adaptability.

## 3. Research Methodology

### 3.1 Research Design

This study adopts a conceptual, architecture-based research design to examine Clean Core Architecture as a foundation for sustainable SAP S/4HANA transformation. A conceptual design is appropriate because the research seeks to integrate fragmented knowledge from ERP customization, enterprise architecture, technical debt, cloud extensibility, and IT governance rather than test a specific causal relationship through primary empirical data. Existing studies show that ERP customization involves interdependent technical and organizational decisions whose long-term consequences cannot be assessed solely through implementation performance indicators (Brehm et al., 2001; Haines, 2009; Strong & Volkoff, 2010).

The architecture-based perspective treats SAP S/4HANA as part of a wider enterprise system comprising the digital core, extension platforms, interfaces, data services, governance structures, and business processes. This perspective reflects the argument that sustainable enterprise architecture depends on standardized foundations, clearly defined system boundaries, and coordinated decision rights (Ross et al., 2006; Weill & Ross, 2004). It also enables the study to assess whether Clean Core mechanisms reduce overall system complexity or merely transfer complexity from the ERP core to external applications and integration layers.

The research design is analytical rather than purely descriptive. It evaluates Clean Core mechanisms against maintainability, upgrade compatibility, flexibility, risk, and technical debt. These criteria address the central tension between organizational differentiation and architectural stability identified in ERP customization research (Luo & Strong, 2004; Rothenberger & Srite, 2009). The approach therefore supports the development of a governance framework linking architectural choices to long-term transformation outcomes.

### 3.2 Literature Selection

The literature was selected through a purposive and concept-oriented process. Sources were included when they contributed directly to at least one of the study's central themes: ERP customization, organizational-system fit, enterprise-system upgrades, technical debt, enterprise architecture, cloud ERP, modular extensibility, digital transformation, or IT governance.

The selected evidence base combines peer-reviewed journal articles, conference studies, scholarly books, SAP technical documentation, and relevant industry reports. Peer-reviewed research provides the theoretical foundations for customization, technical debt, architecture, and governance, while technical documentation supports the interpretation of SAP-specific mechanisms such as released APIs, ABAP Cloud, Embedded Steampunk, and SAP Business Technology Platform extensibility. This combination is necessary because academic ERP research provides strong conceptual explanations but offers limited coverage of recent SAP-specific extensibility models.

The screening process involved three stages. First, sources were assessed for direct relevance to the research problem. Studies addressing unrelated ERP implementation factors without implications for customization, extensibility, upgrades, or governance were excluded. Second, the remaining publications were evaluated for conceptual contribution. Particular attention was given to studies explaining customization choices, system misfit, lifecycle complexity, technical-debt management, and architectural decision rights (Brehm et al., 2001; Li et al., 2015; Strong & Volkoff, 2010; Weill & Ross, 2004). Third, the selected sources were grouped into analytical themes and compared to identify areas of agreement, tension, and incomplete coverage.

The selection process also recognized differences in evidential strength. Systematic and tertiary studies were used to establish broader patterns in cloud ERP and technical-debt research (Abd Elmonem et al., 2016; Li et al., 2015; Rios et al., 2018). Empirical ERP studies were used to interpret customization practices and organizational misfit (Haines, 2009; Hustad et al., 2016; Rothenberger & Srite, 2009). Enterprise architecture and governance studies provided the basis for developing decision rights and accountability mechanisms (Ross et al., 2006; Weill & Ross, 2004).

### 3.3 Analytical Framework

The analytical framework evaluates Clean Core choices through five interrelated criteria. These criteria were selected because they capture both the immediate technical effects and the longer-term architectural consequences of ERP customization.

Maintainability concerns the effort required to understand, modify, test, and support an extension throughout its lifecycle. Technical-debt research demonstrates that tightly coupled and poorly documented solutions increase maintenance burdens and reduce the ability to respond efficiently to future requirements (Li et al., 2015; Tom et al., 2013).

Upgrade compatibility evaluates whether a customization can remain operational when the underlying SAP platform changes. Extensive modifications, internal dependencies, and unsupported interfaces increase remediation and regression-testing requirements during enterprise-system upgrades (Feldman et al., 2016). Clean Core mechanisms are therefore assessed according to their capacity to minimize direct dependency on unstable core objects.

Flexibility refers to the ability to accommodate business change without undermining architectural consistency. ERP research shows that flexibility must be balanced against standardization because unrestricted tailoring may improve short-term fit while creating long-term complexity (Brehm et al., 2001; Luo & Strong, 2004). Side-by-side and modular extensions may improve independent evolution, although distributed architectures can introduce integration and operational challenges (Soldani et al., 2018).

Risk covers architectural, operational, integration, security, and transformation exposure. Cloud and outsourcing research indicates that externally delivered services can provide scalability and flexibility while creating new dependencies involving control, governance, and service continuity (Abd Elmonem et al., 2016; Schneider & Sunyaev, 2016). Each Clean Core mechanism is therefore evaluated according to whether it reduces total transformation risk rather than simply relocating it.

Technical debt examines the future obligations created by present architectural choices. Such debt may include unsupported code, duplicated logic, obsolete interfaces, inadequate documentation, and deferred modernization. Technical debt can be strategically accepted, but it must be visible, measured, prioritized, and governed (Kruchten et al., 2012; Rios et al., 2018).

**Table 1.** Clean Core Evaluation Criteria.

| Criterion | Purpose | Expected Outcome |
|---|---|---|
| Maintainability | Ease of change | Lower support effort |
| Upgrade Compatibility | Release readiness | Reduced remediation |
| Flexibility | Controlled adaptation | Modular innovation |
| Risk | Exposure assessment | Safer transformation |
| Technical Debt | Legacy burden | Lower future cost |

The criteria are applied comparatively rather than independently. An extension may provide greater flexibility while increasing integration risk and maintenance complexity. Therefore, the framework does not assume that one extension model is suitable for every business requirement. Governance maturity functions as a cross-cutting condition because architectural standards, approval controls, monitoring, and accountability influence all five evaluation criteria.

### 3.4 Framework Development

The proposed governance framework was developed through structured concept extraction and synthesis. Concepts identified in the selected literature were first grouped into four categories: architectural principles, customization decisions, lifecycle risks, and governance mechanisms. Relationships were then established between these categories to determine how Clean Core practices could produce sustainable outcomes.

The architectural component was derived from research emphasizing standardization, modularity, stable interfaces, and reusable enterprise capabilities (Ross et al., 2006). The customization component was informed by studies showing that tailoring decisions should reflect business value, organizational fit, implementation effort, and long-term system consequences (Brehm et al., 2001; Parthasarathy & Daneva, 2016). The lifecycle component incorporated maintainability, upgrade effort, technical debt, and integration risk (Feldman et al., 2016; Li et al., 2015). The governance component was based on the allocation of decision rights, accountability, monitoring responsibilities, and architectural ownership (Weill & Ross, 2004).

The framework organizes legacy-customization decisions into five possible treatments: retain, retire, replace, refactor, or externalize. Each customization is assessed against the five analytical criteria before an extension model is selected. Standard SAP functionality is preferred where it adequately supports the requirement. Embedded extensions are considered where close transactional integration is necessary, while side-by-side extensions are evaluated where independent deployment, scalability, or differentiated functionality provides greater architectural value.

The framework also introduces governance controls covering extension approval, released-interface compliance, architecture review, technical-debt registration, documentation, testing, and lifecycle ownership. These controls are intended to prevent Clean Core from becoming a one-time migration activity. Sustainable transformation requires continuous monitoring because new extensions can gradually recreate the complexity removed during the initial SAP S/4HANA transition.

The methodology therefore connects the literature base, analytical criteria, architectural evaluation, and governance design into a coherent research process. The next section applies this framework to Clean Core architectural mechanisms and evaluates their roles, trade-offs, and suitability within SAP S/4HANA environments.

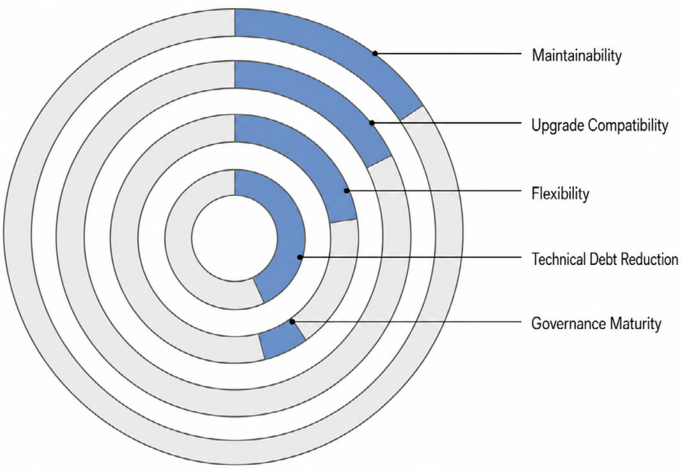


**Figure 2:** Conceptual radial comparison of the five dimensions used to evaluate Clean Core architecture and governance.

## 4. Clean Core Architectural and Extensibility Model

### 4.1 Core Cleanliness and Standardization

Core cleanliness requires SAP S/4HANA implementations to prioritize standard functionality and restrict direct modification of the transactional core. This principle addresses the long-term tension between organizational fit and architectural stability. Although customization may resolve immediate process gaps, excessive tailoring creates dependencies that increase testing, support, and upgrade effort (Brehm et al., 2001; Rothenberger & Srite, 2009). Standardization should therefore be treated as the default architectural position, while deviations require evidence of strategic, regulatory, or operational necessity. Such discipline establishes a stable foundation for business execution and reduces fragmented technical decision-making (Ross et al., 2006).

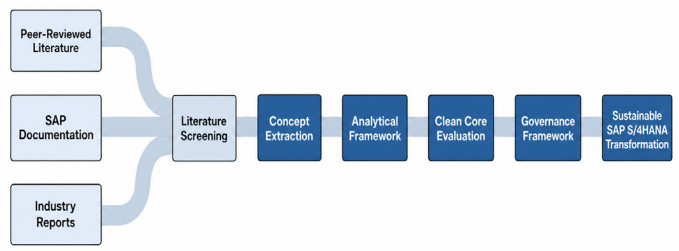


**Figure 3**. Methodological workflow used to synthesize the evidence base, evaluate Clean Core mechanisms, and develop the governance framework for sustainable SAP S/4HANA transformation.

### 4.2 Released APIs and Extension Points

Released APIs and approved extension points provide controlled boundaries between SAP S/4HANA and custom applications. Their architectural value lies in reducing reliance on internal objects that may change during upgrades. This separation improves maintainability by limiting direct coupling and clarifying ownership between standard and custom functionality. However, API adoption alone does not guarantee simplicity. Poorly governed integrations can reproduce complexity through excessive interfaces, duplicated logic, and inconsistent data flows. Consequently, released interfaces must be supported by integration standards, documentation, and lifecycle accountability, consistent with established IT governance principles (Weill & Ross, 2004).

### 4.3 ABAP Cloud Development Model

The ABAP Cloud development model supports controlled extensibility through restricted language use, approved development objects, and released interfaces. Its importance lies in constraining developers to upgrade-stable practices rather than allowing unrestricted access to the underlying system. Such restrictions may initially reduce development freedom, but they improve transparency and lower future remediation requirements. Technical-debt research demonstrates that unmanaged dependencies and short-term design choices frequently create significant lifecycle costs (Kruchten et al., 2012; Li et al., 2015). ABAP Cloud therefore functions as both a development model and a preventive technical-debt mechanism.

### 4.4 Embedded Steampunk

Embedded Steampunk applies the ABAP Cloud model within the SAP S/4HANA environment, enabling extensions that remain close to transactional processes while following controlled development rules. It is particularly relevant where low latency, direct business-object interaction, or tightly integrated application behaviour is required. Nevertheless, embedded development should not become a substitute for architectural evaluation. Retaining too much custom logic within the system may preserve dependencies that Clean Core seeks to reduce. Embedded Steampunk is therefore most appropriate when transactional proximity provides clear value and the extension can remain isolated from unreleased core objects.

### 4.5 Side-by-Side Extensions on SAP BTP

Side-by-side extensibility on SAP Business Technology Platform separates differentiated functionality from the ERP core. This approach supports independent deployment, cloud-native development, scalability, and integration with external services. Its modularity aligns with broader architectural movements toward loosely coupled services, although distributed systems may introduce monitoring, security, integration, and operational complexity (Soldani et al., 2018). Side-by-side extensions should therefore be selected when independent evolution outweighs additional integration overhead.

Overall, the Clean Core model requires deliberate selection among standard functionality, released APIs, embedded extensions, and SAP BTP services. The next section examines how legacy customizations can be retired, replaced, refactored, retained, or externalized within this architecture.

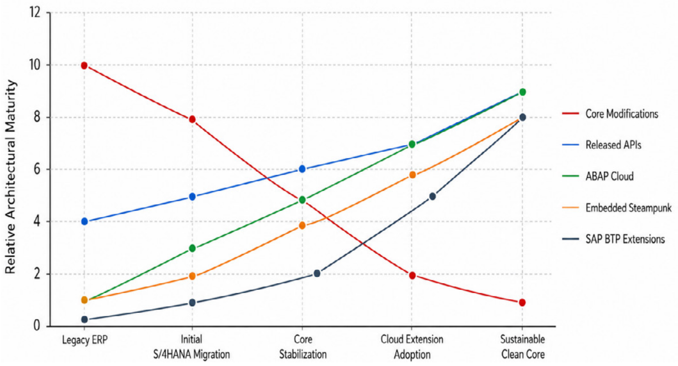


**Figure 4:** Conceptual development of SAP extensibility approaches from modification-intensive legacy environments toward sustainable Clean Core architecture.

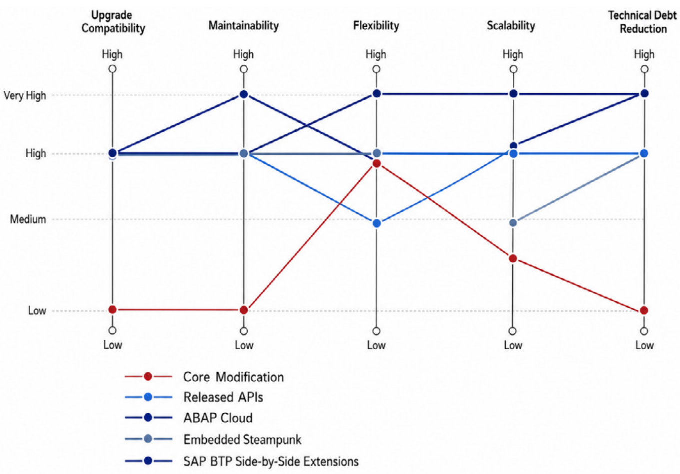


**Figure 5:** Comparative evaluation of SAP extensibility approaches across upgrade compatibility, maintainability, flexibility, scalability, and technical-debt dimensions.

## 5. Migration of Legacy Customizations

Migration of legacy customizations should begin with a structured assessment of business value, usage, technical condition, architectural coupling, and upgrade impact. Custom developments that once resolved process misfits may no longer justify their maintenance burden, particularly where standard SAP S/4HANA functionality can meet the underlying requirement. ERP customization research shows that tailoring decisions frequently reflect historical organizational preferences rather than enduring strategic necessity, creating dependencies that complicate future transformation (Brehm et al., 2001; Haines, 2009; Hustad et al., 2016). Each customization should therefore be classified for retirement, replacement, refactoring, or retention based on its current value and lifecycle risk.

Retirement is appropriate for unused or redundant developments, while replacement should be preferred where standard functionality provides comparable outcomes. Refactoring is justified when differentiated business logic remains valuable but depends on obsolete objects or unsupported interfaces. Retention should be limited to customizations that are operationally critical, technically stable, and economically defensible. Requirement prioritization is essential because preserving every historical customization increases implementation effort and weakens architectural coherence (Parthasarathy & Daneva, 2016).

Where custom logic remains necessary, migration to released APIs and ABAP Cloud can reduce dependence on internal SAP objects and improve upgrade compatibility. This transition converts unrestricted custom code into governed extensions with clearer boundaries and lower remediation exposure. However, externalization should not be automatic. SAP BTP side-by-side extensions are most suitable where independent deployment, scalability, cloud-native services, or cross-system integration provide material advantages. Although modular architectures improve flexibility, they may also introduce distributed monitoring, integration, and operational complexity (Soldani et al., 2018).

Migration risks include data inconsistency, interface failure, security exposure, performance degradation, and business disruption. Mitigation therefore requires dependency mapping, staged testing, access controls, rollback planning, documentation, and clearly assigned lifecycle ownership. Upgrade and transformation studies confirm that technical change must be coordinated with organizational readiness and operational continuity (Feldman et al., 2016). Effective migration consequently reduces technical debt only when modernization decisions are governed, documented, and continuously reviewed (Li et al., 2015; Rios et al., 2018). The next section evaluates the sustainability benefits and governance controls required to preserve these outcomes.

## 6. Sustainability Benefits and Governance Framework

The sustainability value of Clean Core Architecture depends on its capacity to reduce technical debt while preserving upgrade compatibility and business adaptability. Custom code, unreleased dependencies, and duplicated logic create future maintenance obligations that increase testing, remediation, and support effort (Kruchten et al., 2012; Li et al., 2015). Clean Core practices reduce this burden by prioritizing standard functionality, released APIs, controlled extensions, and systematic retirement of obsolete developments. However, technical debt is not eliminated automatically; it may reappear through poorly governed SAP BTP applications, fragmented integrations, or undocumented extensions. Sustainable outcomes therefore require continuous identification, prioritization, and repayment of architectural debt (Rios et al., 2018).

Upgrade compatibility improves when extensions are separated from unstable core objects and designed through supported interfaces. This reduces regression-testing scope and lowers the organizational disruption associated with enterprise-system upgrades (Feldman et al., 2016). Nevertheless, upgrade readiness should be treated as a governance outcome rather than a purely technical attribute. The proposed Clean Core governance framework combines architectural standards, extension approval, released-interface compliance, lifecycle ownership, security review, documentation, and periodic reassessment. Such controls align with enterprise architecture principles emphasizing standardized foundations and clearly assigned decision rights (Ross et al., 2006; Weill & Ross, 2004).

A customization decision board should coordinate business owners, enterprise architects, developers, security specialists, and operations teams. Its role is to evaluate proposed extensions against business value, strategic differentiation, technical risk, maintainability, and total lifecycle cost. This collective structure limits locally optimized decisions that may produce enterprise-wide complexity, a recurrent weakness in ERP customization practice (Brehm et al., 2001; Rothenberger & Srite, 2009).

Governance effectiveness should be monitored through measurable indicators, including total core modification count, proportion of integrations using released APIs, upgrade remediation effort, unresolved technical-debt backlog, extension retirement rate, and architecture-compliance exceptions. These indicators make architectural deterioration visible and enable corrective action before complexity becomes embedded. Consequently, Clean Core governance reduces transformation risk by converting architectural discipline into an ongoing operating capability. The concluding section synthesizes these findings and evaluates their broader implications for sustainable SAP S/4HANA transformation.

## 7. Discussion

The findings indicate that Clean Core Architecture can improve long-term ERP sustainability, but only when architectural complexity is reduced rather than transferred from SAP S/4HANA to SAP BTP and external services. Standardization, released APIs, and controlled extensibility can decrease core dependencies, remediation effort, and upgrade disruption. However, side-by-side applications may introduce distributed integration, monitoring, security, data-consistency, and operational challenges comparable to those identified in modular and microservice environments (Soldani et al., 2018). Cloud-based extensibility may therefore replace modification debt with integration or governance debt when services are fragmented, undocumented, or weakly controlled (Abd Elmonem et al., 2016; Schneider & Sunyaev, 2016). Clean Core should consequently be evaluated through total architectural complexity rather than the cleanliness of the transactional core alone.

The study contributes theoretically by connecting ERP customization research with enterprise architecture and technical debt management. Earlier studies establish that customization can improve organizational fit while creating dependencies that constrain system evolution (Brehm et al., 2001; Strong & Volkoff, 2010). This study extends that argument by positioning Clean Core as a lifecycle governance model in which debt prevention, extension placement, and upgrade compatibility are treated as interdependent architectural decisions (Kruchten et al., 2012; Li et al., 2015). It also contributes to digital transformation literature by showing that sustainable modernization depends on foundational architecture and governance, not merely technology adoption (Vial, 2019; Verhoef et al., 2021).

For CIOs and business leaders, the findings emphasize the need to challenge customization demands against strategic value and lifecycle cost. Enterprise architects and SAP consultants must select extension models according to coupling, scalability, security, and maintainability, while customization boards should enforce decision rights and accountability (Ross et al., 2006; Weill & Ross, 2004). These implications provide the basis for the concluding assessment of Clean Core as a sustainable transformation strategy.

## 8. Conclusion

This study establishes Clean Core Architecture as a strategic foundation for sustainable SAP S/4HANA transformation rather than a narrowly technical approach to limiting customization. The analysis shows that long-term value depends on maintaining a stable digital core, using released interfaces, selecting appropriate embedded or side-by-side extension models, and systematically addressing legacy customizations according to their business value and architectural risk. Clean Core can reduce technical debt, improve upgrade compatibility, strengthen maintainability, and enable controlled innovation, but these outcomes are not automatic. They depend on coordinated enterprise architecture, disciplined migration strategies, effective governance, and continuous monitoring across the ERP lifecycle.

The proposed governance framework contributes academically by integrating customization management, technical debt control, extensibility decisions, and upgrade readiness within a unified enterprise architecture perspective. Practically, it provides organizations with a structured basis for extension approval, lifecycle ownership, architectural compliance, and performance measurement. The conceptual nature of the study limits direct generalization across different SAP environments. Future research should therefore validate the framework through comparative case studies of real-world SAP S/4HANA implementations. Ultimately, Clean Core sustainability is achieved not by eliminating customization, but by governing it as a long-term architectural responsibility.

## 9. References


1. Abd Elmonem MA, Nasr ES, Geith MH. Benefits and challenges of cloud ERP systems-A systematic literature review. Future Computing and Informatics Journal, 2016;1(1–2):1-9.
2. Brehm L, Heinzl A, Markus ML. Tailoring ERP systems: A spectrum of choices and their implications. Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2001:1-9.
3. Cao R, Iansiti M. Digital transformation, data architecture, and legacy systems. Journal of Digital Economy, 2022;1(1):1-19.
4. Feldman G, Shah H, Chapman C, Amini A. Enterprise systems: The upgrade process model. Journal of Enterprise Information Management, 2016;29(6):822-840.
5. Potla RB. ISCSITR–International Journal of ERP and CRM (ISCSITR. IJEC), 2022;3(1):11-38.
6. Potla RB. Value-Stream-Aligned Order-to-Cash in SAP S/4HANA: Lead-Time Compression with ATP, BOP and aTO/Pegging Strategies, 2021;10.
7. Haines MN. Understanding enterprise system customization: An

exploration of implementation realities and the key influence factors. Information Systems Management, 2009;26(2):182-198.

8. Hustad E, Haddara M, Kalvenes B. ERP and organizational misfits: An ERP customization journey. Procedia Computer Science, 2016;100:429-439.

9. Johansson B, Alajbegovic A, Alexopoulo V, Desalermos A. Cloud ERP adoption opportunities and concerns: The role of organizational size. In: 2015 48th Hawaii International Conference on System Sciences, 2015:4211-4219.

10. Kruchten P, Nord RL, Ozkaya I. Technical debt: From metaphor to theory and practice. IEEE Software, 2012;29(6):18-21.

11. Potla RB. Blueprinting a manufacturing data lakehouse: Harmonizing BOM, routing, and serialization data for advanced analytics. International Journal of Innovative Research in Engineering & Multidisciplinary Physical Sciences, 2021;9(1):1-12.

12. Li Z, Avgeriou P, Liang P. A systematic mapping study on technical debt and its management. Journal of Systems and Software, 2015;101:193-220.

13. Luo W, Strong DM. A framework for evaluating ERP implementation choices. IEEE Transactions on Engineering Management, 2004;51(3):322-333.

14. Parthasarathy S, Daneva M. An approach to estimation of degree of customization for ERP projects using prioritized requirements. Journal of Systems and Software, 2016;117:471-487.

15. Rios N, de Mendonça Neto MG, Spínola RO. A tertiary study on technical debt: Types, management strategies, research trends, and base information for practitioners. Information and Software Technology, 2018;102:117-145.

16. Ross JW, Weill P, Robertson D. Enterprise architecture as strategy: Creating a foundation for business execution. Harvard Business Press, 2006.

17. Rothenberger MA, Srite M. An investigation of customization in ERP system implementations. IEEE Transactions on Engineering Management, 2009;56(4):663-676.

18. Schneider S, Sunyaev A. Determinant factors of cloud-sourcing decisions: Reflecting on the IT outsourcing literature in the era of cloud computing. Journal of Information Technology, 2016;31(1):1-31.

19. Soldani J, Tamburri DA, van den Heuvel W-J. The pains and gains of microservices: A systematic grey literature review. Journal of Systems and Software, 2018;146:215-232.

20. Strong DM, Volkoff O. Understanding organization–enterprise system fit: A path to theorizing the information technology artifact. MIS Quarterly, 2010;34(4):731-756.

21. Tom E, Aurum A, Vidgen R. An exploration of technical debt. Journal of Systems and Software, 2013;86(6):1498-1516.

22. Uppström E, Lönn C-M, Hoffsten M, Thorström J. New implications for customization of ERP systems. In: 2015 48th Hawaii International Conference on System Sciences, 2015:4220-4229.

23. Verhoef PC, Broekhuizen T, Bart Y, Bhattacharya A, Dong JQ, Fabian N, Haenlein M. Digital transformation: A multidisciplinary reflection and research agenda. Journal of Business Research, 2021;122:889-901.

24. Vial G. Understanding digital transformation: A review and a research agenda. The Journal of Strategic Information Systems, 2019;28(2):118-144.

25. Weill P, Ross JW. IT governance: How top performers manage IT decision rights for superior results. Harvard Business Press, 2004.