International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

To cite this article: Kasyapp Ivaaturi (2026). MODERN TRENDS IN THE DEVELOPMENT OF ENTERPRISE APPLICATION ARCHITECTURE FOR DISTRIBUTED TEAMS, International Journal of Research in Commerce and Management Studies (IJRCMS) 8 (2): 151-165 Article No. 675 Sub Id 1148

MODERN TRENDS IN THE DEVELOPMENT OF ENTERPRISE APPLICATION ARCHITECTURE FOR DISTRIBUTED TEAMS

Kasyapp Ivaaturi

VP of Enterprise Applications
London, UK

DOI: https://doi.org/10.38193/IJRCMS.2026.8213

ABSTRACT

The article is dedicated to the analysis of modern trends in the development of enterprise application architecture under conditions of distributed teamwork. The relevance of the study is determined by the rapid expansion of geographically dispersed software teams, the intensification of inter-team dependencies, and the increasing complexity of architectural governance in large-scale digital systems. The novelty of the research lies in the interpretation of enterprise architecture not merely as a technical framework but as a coordination regime shaped by scaling dynamics, ownership structures, platform engineering practices, and modernization trajectories. The work describes structural transformations associated with microservices adoption, legacy system modernization, platform consolidation, and remote collaboration. Special attention is paid to the relationship between modularization and coordination density, as well as to the tension between autonomy and synchronization in distributed environments. The goal of the study is to identify systemic regularities that determine architectural evolution in distributed enterprise systems. Comparative and analytical methods were applied to synthesize contemporary research findings. The conclusions characterize architectural development as cyclical and governance-dependent. The article will be useful for enterprise architects, IT managers, and researchers studying large-scale distributed software systems.

KEYWORDS: enterprise architecture, distributed teams, microservices, platform engineering, legacy modernization, coordination mechanisms, governance models, remote collaboration

INTRODUCTION

The digital transformation of enterprises has fundamentally altered the structural logic of application architecture. Modern organizations increasingly rely on distributed teams operating across geographical, organizational, and temporal boundaries. This structural shift intensifies coordination complexity and redefines architectural decision-making. Architectural decomposition, once primarily a technical concern, is now inseparable from ownership distribution, governance configuration, and synchronization requirements.

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

The relevance of this research is associated with the growing discrepancy between traditional architectural models and the coordination realities of distributed software development. Microservice adoption, platform engineering initiatives, and hybrid modernization strategies have expanded architectural flexibility, yet have simultaneously introduced new dependency patterns. Under these conditions, enterprise architecture evolves as a response to coordination pressure rather than as a purely technological choice.

The purpose of this study is to develop a systemic understanding of how enterprise application architecture adapts to distributed team structures and to identify mechanisms enabling effective coordination. To achieve this purpose, three tasks are formulated:

1. Assess the influence of team scaling and coordination density on architectural modularization.
2. Evaluate the impact of governance redistribution and platform engineering on distributed system stability.
3. Identify cyclical patterns of architectural evolution in distributed enterprise systems.

This study interprets enterprise architecture as a dynamic coordination regime, integrating technical, managerial, and infrastructural perspectives. We hypothesize that the effectiveness of distributed enterprise architecture is maximized when coordination mechanisms are aligned with ownership distribution and modular boundaries.

METHODS AND MATERIALS

The research is based on a comprehensive analysis of contemporary scholarly works devoted to distributed software development, enterprise architecture transformation, and coordination mechanisms in large-scale systems.

Berntzen et al. investigated how coordination mechanisms evolve in large-scale agile environments over time and demonstrated that structural scaling leads to measurable changes in coordination configurations (Berntzen et al., 2023). Li et al. analyzed microservice ownership patterns in open-source ecosystems and revealed persistent organizational coupling despite technical decomposition (Li et al., 2025). Alzoubi and Mishra examined the contribution of enterprise architecture frameworks in distributed agile software development and emphasized their role in achieving alignment across dispersed teams (Alzoubi and Mishra, 2024). Makris et al. studied the transition from monolithic to microservice-based applications and identified technical, cognitive, and organizational constraints affecting decomposition strategies (Makris et al., 2022). Abd-Elwahab et al. proposed a microservices-driven enterprise architecture model focused on infrastructure optimization and demonstrated efficiency gains through alignment of deployment granularity and responsibility domains (Abd-

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

Elwahab et al., 2023). Dursun analyzed the transition toward platform engineering and described the shift from externally integrated capabilities to embedded infrastructural standards (Dursun, 2023). Nguyen-Duc et al. conducted a global study on work-from-home practices and examined how distributed collaboration reshapes coordination routines and architectural documentation intensity (Nguyen-Duc et al., 2024). Martínez et al. explored microservice-based architectures in real-time industrial applications and demonstrated the necessity of deterministic orchestration layers under latency constraints (Martínez et al., 2026). Fávero et al. carried out a systematic mapping study on legacy system modernization and identified hybrid coexistence strategies as dominant transitional models (Fávero et al., 2025).

To write the article, a comparative method, structural analysis, synthesis of empirical findings, and interpretive modeling were applied. The materials were analyzed to identify convergent trends, structural tensions, and recurrent transformation patterns across distributed enterprise contexts.
The combination of these analytical approaches ensured a comprehensive examination of architectural evolution under distributed development conditions.

RESULTS

Distributed enterprise application architecture is no longer shaped primarily by technological preference; it is reorganized by coordination pressure. As teams multiply, disperse geographically, and interact with heterogeneous stakeholders, architectural decisions increasingly encode assumptions about communication latency, ownership boundaries, and synchronization rhythms. The architectural configuration becomes a response to inter-team dependency density rather than an abstract design ideal.

A first trajectory concerns the interaction between organizational scaling and coordination mechanisms. Longitudinal evidence collected across 62 days of fieldwork, 37 interviews, and 118 observed meetings over 1.5 years demonstrates that coordination arrangements evolve in parallel with structural growth, not after it (Berntzen et al., 2023). External events, such as onboarding new clients, generated additional cross-team dependencies that required new coordination roles and shared artifacts. The number of teams increased from five to 17 within a few years, and the architectural landscape shifted accordingly toward more explicit inter-team interfaces and formalized technical forums (Berntzen et al., 2023). The evidentiary signal is unambiguous: architectural modularization intensified as coordination mechanisms multiplied.

Yet scaling does not automatically justify complexity. When a large-scale program transitioned from a mixed first-generation approach to cross-functional autonomous teams with continuous delivery, the number of coordination mechanisms decreased from 27 to 14 while perceived coordination

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

effectiveness increased (Berntzen et al., 2023). The reduction did not simplify architecture in a superficial sense; it reorganized dependency management around clearer ownership boundaries. Architecture here acts as a compression device: fewer coordination mechanisms, sharper structural demarcations.

Ownership distribution across microservices reveals a second structural trajectory. Empirical analysis of open-source ecosystems shows that microservice ownership frequently clusters around small groups, generating implicit organizational coupling even when technical coupling appears minimized (Li et al., 2025). Organizational proximity persists in code contribution patterns, and microservice autonomy remains contingent on communication topology rather than repository boundaries (Li et al., 2025). The mechanism is subtle: architectural decoupling without governance redesign produces hidden synchronization costs.

Enterprise architecture interventions in distributed agile settings reinforce this observation. Analytical modeling indicates that enterprise architecture contributes to alignment by stabilizing shared abstractions across dispersed teams, reducing interpretive divergence in large-scale coordination (Alzoubi and Mishra, 2024). Architectural artifacts function as negotiation substrates; they do not eliminate coordination effort but reallocate it.

Migration from monolithic systems introduces another layer of friction. Developers report that decomposition decisions are rarely determined solely by functional segmentation; they are constrained by legacy data entanglement, team skill asymmetry, and release orchestration challenges (Makris et al., 2022). The move from monolith to microservices produces not only service boundaries but also new cognitive boundaries. Coordination cost is redistributed, not removed.

Infrastructure optimization driven by microservices-oriented enterprise architecture reveals quantifiable efficiency improvements when infrastructure components are aligned with service granularity (Abd-Elwahab et al., 2023). Resource utilization patterns become more predictable once deployment units correspond to organizational responsibility clusters. Architectural constraint, in this case, reorganizes cost structures through alignment between technical and managerial segmentation.

Platform engineering introduces an additional structural pivot. The shift from “bolting-on” to “building-in” quality attributes repositions infrastructure capabilities as shared internal platforms rather than afterthought integrations (Dursun, 2023). This reconfiguration alters team coordination patterns: developers negotiate feature logic within predefined platform affordances, reducing variance in deployment and configuration practices. Architectural standardization here does not centralize authority; it embeds governance into reusable platform modules.

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

Remote collaboration intensifies these dynamics. A global study identifies altered stakeholder perceptions and modified development practices under work-from-home conditions (Nguyen-Duc et al., 2024). Coordination becomes more artifact-driven and less reliant on synchronous interaction. Architectural documentation density increases as informal clarification channels weaken. Distributed settings amplify the value of explicit interface contracts.

Real-time automation requirements in smart industries push microservice architectures toward stricter latency constraints. An end-to-end architecture designed to satisfy real-time requirements demonstrates that service orchestration layers must incorporate deterministic communication pathways to meet timing guarantees (Martínez et al., 2026). Distributed teams coordinating such systems cannot rely on loosely coupled asynchronous flows alone; architectural layering becomes temporally disciplined.

Legacy modernization efforts further illuminate the evolutionary pattern. A systematic mapping study documents modernization strategies that balance incremental refactoring with full service extraction (Fávero et al., 2025). Hybrid coexistence models appear frequently, where monolithic cores persist alongside emergent microservices. Architecture evolves through staged detachment rather than abrupt replacement. The modernization process itself becomes a coordination mechanism, sequencing technical transformation with organizational learning. Across these trajectories, several structural regularities emerge. The systematization of architectural transformation mechanisms is presented below (Table 1).

Table 1. Structural transformations in enterprise application architecture for distributed teams (compiled by the author based on Berntzen et al., 2023; Li et al., 2025; Alzoubi and Mishra, 2024; Makris et al., 2022; Abd-Elwahab et al., 2023; Dursun, 2023; Nguyen-Duc et al., 2024; Martínez et al., 2026; Fávero et al., 2025)

Transformation driver	**Architectural response**	**Coordination implication**	**Governance adjustment**
Organizational scaling	Modularization and service boundary clarification	Increase in inter-team interfaces	Formalization of ownership domains

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

Reduction of coordination mechanisms	Structural compression of interfaces	Higher clarity of dependency topology	Selective elimination of redundant processes
Microservice ownership clustering	Service-level decoupling with latent coupling	Informal coordination persistence	Redistribution of decision authority
Monolith decomposition	Incremental service extraction	Redistribution of cognitive load	Phased governance realignment
Platform engineering adoption	Internal platform layer standardization	Reduced feature-level negotiation	Embedded infrastructural control
Remote collaboration	Interface formalization and artifact density growth	Artifact-mediated coordination	Documentation-driven alignment
Real-time constraints	Deterministic orchestration layers	Reduced asynchronous flexibility	Centralized temporal governance
Legacy coexistence	Hybrid architecture models	Dual coordination logics	Transitional control regimes

First, architectural modularity correlates with coordination density. As the number of interdependent teams increases, modularization intensifies, but only when ownership clarity is reinforced by governance instruments. Absent governance realignment, microservices multiply without reducing coupling.

Second, coordination mechanisms fluctuate in number and form, yet architectural coherence depends less on quantity than on temporal alignment. A program that reduced coordination mechanisms from 27 to 14 achieved higher effectiveness because mechanisms were synchronized with dependency patterns (Berntzen et al., 2023). Architecture functions optimally when coordination instruments match dependency topology.

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

Third, platform engineering stabilizes distributed development by relocating variability into shared infrastructure layers. Teams adjust feature implementation within bounded architectural constraints. The institutional arrangement becomes more resilient as platform capabilities internalize cross-cutting concerns.

Fourth, remote work conditions alter the relative weight of documentation, automation, and interface formalization. Artifact-mediated coordination gains prominence, and architectural traceability improves as teams compensate for reduced spontaneous interaction. The shift is not merely behavioral; it restructures architectural artifact ecosystems.

Fifth, modernization trajectories rarely eliminate legacy constraints entirely. Hybrid states dominate transitional phases. These hybrid architectures require dual coordination logics: one for stable legacy modules, another for rapidly evolving microservices. Tension persists.

Finally, real-time industrial applications demonstrate that distributed microservice architectures can satisfy strict timing constraints only when orchestration layers are explicitly engineered for deterministic behavior. Architectural freedom narrows under latency requirements. Distributed autonomy coexists with centralized temporal discipline.

These findings converge on a broader analytical observation: enterprise application architecture for distributed teams evolves through cycles of expansion, simplification, and stabilization. The structural cycle of architectural evolution is illustrated below (Figure 1).

Figure 1. Cyclical evolution of enterprise application architecture in distributed teams (compiled by the author based on Berntzen et al., 2023; Li et al., 2025; Dursun, 2023; Fávero et al., 2025)

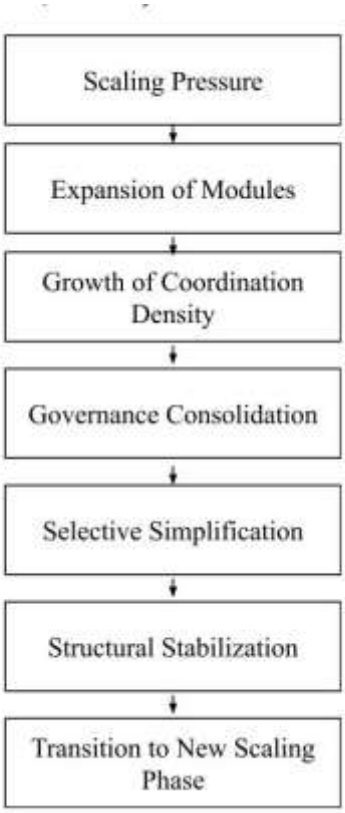
Coordination crises trigger structural adjustments; stabilization phases consolidate governance patterns; subsequent scaling reintroduces complexity. The movement is iterative. It never settles.

Quantitative evidence reinforces this pattern. Observations spanning 62 days and 118 meetings reveal continuous modification of coordination roles and tools (Berntzen et al., 2023). The decrease from 27 to 14 coordination mechanisms during structural transition indicates that architectural effectiveness is not proportional to managerial instrumentation (Berntzen et al., 2023). The persistence of clustered ownership in microservice ecosystems signals that technical decomposition alone does not dissolve organizational coupling.

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165



Enterprise architecture contributions documented in distributed agile settings illustrate measurable alignment gains when architectural frameworks guide cross-team integration. Microservices-driven infrastructure models demonstrate cost and optimization benefits once architectural segmentation mirrors operational responsibility units.

At the same time, developers navigating monolith decomposition report sustained cognitive load during transitional states. Platform engineering initiatives indicate that embedding quality constraints within shared platforms reduces downstream coordination variability. Work-from-home environments reshape stakeholder expectations and amplify reliance on structured artifacts.

Architecture, therefore, operates simultaneously as a technical blueprint, governance scaffold, and coordination instrument. The differentiation of architectural functions in distributed enterprise systems is presented below (Table 2).

Table 2. Functional dimensions of enterprise application architecture in distributed teams (compiled by the author based on Alzoubi and Mishra, 2024; Li et al., 2025; Dursun, 2023; Nguyen-Duc et al.,

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

2024; Martínez et al., 2026; Fávero et al., 2025)

Architectural dimension	Primary structural focus	Coordination effect	Risk if underdeveloped
Structural (modularity)	Service boundaries and decomposition logic	Clarifies inter-team responsibility zones	Hidden coupling and synchronization overhead
Governance (ownership)	Decision authority distribution	Reduces ambiguity in prioritization	Centralized bottlenecks despite technical decoupling
Platform (infrastructure layer)	Shared internal services and standards	Stabilizes deployment and quality practices	Configuration drift and environment inconsistency
Informational (artifacts and documentation)	Interface contracts and traceability	Supports asynchronous coordination	Misalignment under remote collaboration
Temporal (delivery rhythm)	Release cadence and orchestration control	Synchronizes distributed contributions	Latency conflicts and integration instability
Transitional (modernization logic)	Hybrid coexistence management	Enables staged architectural evolution	Fragmented integration between legacy and services

Its evolution reflects distributed team dynamics as much as technological progress. One boundary remains visible. As coordination instruments multiply, practitioners report stress associated with constant adjustment. Continuous improvement may erode stability when rhythm exceeds absorption capacity. The architectural configuration must absorb change without overwhelming contributors.

Modern enterprise application architecture for distributed teams does not converge toward a single dominant pattern. It oscillates between modular expansion and governance consolidation, between

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

autonomy and synchronization. The structural tension persists.

DISCUSSION

Distributed enterprise architecture emerges not as a static blueprint but as an evolving coordination regime shaped by fluctuating dependency density. The results demonstrate that architectural transformation in distributed teams rarely follows a purely technological trajectory. Instead, structural change is triggered by shifts in ownership boundaries, scaling velocity, governance intensity, and synchronization demands. What appears at first glance as a transition toward microservices, platform engineering, or modernization frameworks reveals a deeper reorganization of inter-team interaction patterns. Architecture records these shifts. It does not precede them.

A central tension concerns the relationship between modular decomposition and organizational coupling. Technical segmentation into microservices is frequently interpreted as a strategy for autonomy. Yet empirical patterns indicate that service boundaries alone do not dissolve coupling; coordination clusters often persist around informal ownership constellations. This suggests that microservice autonomy depends less on code separation than on redistribution of decision authority. Where governance remains centralized, decomposition may fragment repositories while preserving coordination bottlenecks. Architectural fragmentation without governance recalibration produces distributed opacity rather than distributed independence.

The observed reduction in coordination mechanisms during transitions toward cross-functional, continuous delivery–oriented structures introduce another interpretive friction. Simplification coincided with increased perceived effectiveness. The mechanism underlying this shift appears to involve alignment between dependency topology and coordination instruments. When coordination structures exceed actual dependency needs, redundancy accumulates, producing friction in planning and reporting cycles. When mechanisms are selectively removed, architectural clarity intensifies. This reframes architectural maturity not as an accumulation of processes but as calibrated subtraction.

Scaling introduces a contradictory dynamic. Growth in team count amplifies knowledge, process, and resource dependencies, generating pressure for additional alignment structures. However, expansion of coordination mechanisms beyond absorption capacity produces cognitive overload and institutional fatigue. Distributed teams operating under continuous structural adjustment risk instability if modification cycles accelerate beyond adaptation thresholds. Architectural governance must therefore regulate tempo, not only direction. Stability requires intervals of consolidation between waves of change.

Platform engineering offers one pathway toward structural stabilization. By embedding quality

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

constraints, deployment logic, and infrastructural services into shared internal platforms, variability is relocated from operational coordination into architectural infrastructure. This relocation reduces negotiation overhead at the feature level while increasing the significance of platform design decisions. A new dependency concentration emerges at platform boundaries. Teams gain autonomy within bounded technical affordances, yet collective resilience depends on platform coherence. The locus of coordination shifts upward, from inter-team synchronization to infrastructural stewardship.

Remote collaboration intensifies artifact-mediated coordination. Under distributed work conditions, spontaneous clarification diminishes, and documentation density expands. Architectural traceability, interface formalization, and explicit dependency mapping gain prominence. This trend reinforces the argument that enterprise architecture in distributed contexts functions as a communication instrument as much as a structural diagram. Architecture becomes a durable proxy for synchronous interaction. The absence of physical proximity elevates the epistemic function of architectural artifacts.

Real-time industrial systems complicate this trajectory further. Microservice flexibility, often associated with asynchronous communication and loose coupling, confronts deterministic timing requirements. To satisfy strict latency constraints, orchestration layers must enforce disciplined sequencing and predictable execution paths. Distributed autonomy narrows where temporal guarantees dominate. Architectural decentralization coexists with centralized temporal control. This interplay reveals that distributed enterprise architecture is not uniformly trending toward greater looseness; constraint intensifies in environments governed by real-time performance demands.

Legacy modernization presents another structural paradox. Incremental migration strategies frequently maintain hybrid coexistence between monolithic cores and newly extracted services. Such hybrid states produce dual coordination logics: stable release cycles for legacy components and iterative deployment rhythms for microservices. The architectural system oscillates between stability and experimentation. Complete detachment rarely occurs in a single movement. Instead, modernization unfolds through staged segmentation, each stage recalibrating coordination expectations. Architectural evolution proceeds through layered accumulation rather than abrupt replacement.

Enterprise architecture frameworks contribute interpretive coherence in distributed agile environments by stabilizing shared abstractions across dispersed teams. Yet their contribution depends on flexibility. Rigid frameworks risk ossification; overly adaptive ones dissolve into symbolic guidance. The productive zone lies in adaptive formalization: architectural standards sufficiently structured to reduce ambiguity, yet sufficiently porous to accommodate local adaptation. The friction between standardization and experimentation cannot be eliminated. It must be managed.

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

The results collectively point toward a cyclical movement in distributed enterprise architecture: expansion of modular boundaries, consolidation of governance mechanisms, subsequent reduction of excess coordination, and renewed expansion under scaling pressure. Each cycle leaves residual structures that shape subsequent adaptation. Architectural configuration, therefore, reflects historical layering. Decisions made under one dependency regime persist as sediment in later stages.

A further interpretive boundary concerns human absorption capacity. Continuous improvement cultures encourage experimentation with coordination roles, meetings, tools, and governance routines. Yet sustained adjustment generates fatigue when rhythm exceeds cognitive bandwidth. Distributed teams confronted with overlapping reforms may experience reduced engagement. Architectural change must therefore account for institutional tempo. Structural agility without temporal discipline destabilizes participation.

The interplay between autonomy and synchronization remains unresolved. Distributed teams seek local decision authority, while enterprise architecture demands system-wide coherence. Microservices promise independence, yet cross-cutting concerns—security, compliance, data integrity, latency guarantees—require coordinated enforcement. Platform engineering attempts to reconcile this contradiction by embedding governance within reusable layers. Even so, complete autonomy remains constrained by systemic integration requirements.

Another unresolved tension concerns visibility. As service counts increase, architectural landscapes become opaque. Observability tooling, dependency mapping, and interface documentation expand to restore transparency. Yet greater visibility generates additional data flows, which themselves require interpretation and governance. Transparency scales, but so does interpretive complexity. The coordination rhythm accelerates.

From a methodological perspective, the findings suggest that enterprise architecture research in distributed settings cannot isolate technical variables from organizational processes. Structural decisions are entangled with governance models, ownership configurations, and coordination practices. Analytical separation of architecture from coordination obscures causal interdependence. Architecture shapes coordination; coordination reshapes architecture. The relationship is recursive.

No single architectural paradigm dominates contemporary distributed enterprise development. Microservices, platform engineering, hybrid modernization, and deterministic orchestration coexist as responses to distinct dependency configurations. The apparent diversity reflects adaptive variation rather than conceptual fragmentation. Each pattern addresses specific coordination pressures under particular scaling and governance conditions.

International Journal of Research in Commerce and Management Studies

ISSN 2582-2292

Vol. 8, No. 02 March-April; 2026 Page. No. 151-165

What persists across cases is structural oscillation. Distributed enterprise architecture advances through successive adjustments rather than linear progression. Modularity intensifies until coordination cost accumulates; governance consolidates until flexibility erodes; simplification restores clarity until scaling reintroduces complexity. The movement is iterative, layered, and historically conditioned.

The discussion, therefore, exposes a structural equilibrium that remains provisional. Architectural stability in distributed teams depends on calibrated coordination density, adaptive governance boundaries, infrastructural embedding of shared constraints, and disciplined temporal pacing of change. Remove any of these elements, and oscillation destabilizes. Retain them rigidly, and adaptability contracts.

Enterprise application architecture for distributed teams operates within this narrow corridor between rigidity and diffusion. The corridor shifts with every expansion, every modernization phase, every platform consolidation, every new dependency introduced by scaling. The architecture evolves. It does not settle.

CONCLUSION

The conducted analysis confirms that enterprise application architecture in distributed teams develops under the decisive influence of coordination density, governance redistribution, and infrastructural consolidation. The first task demonstrated that architectural modularization intensifies in response to scaling dynamics, yet its effectiveness depends on clarity of ownership boundaries. The second task revealed that platform engineering, modernization strategies, and enterprise architecture frameworks function as stabilizing instruments only when governance structures adapt simultaneously with technical segmentation. The third task established that architectural transformation follows a cyclical trajectory characterized by expansion, simplification, stabilization, and renewed scaling pressure.

The study confirms that effective enterprise application architecture in distributed teams depends on calibrated coordination density, clear ownership distribution, and embedded infrastructural constraints. Our hypothesis regarding alignment between modularization and coordination mechanisms is supported.

Enterprise application architecture for distributed teams, therefore, operates within a dynamic equilibrium shaped by oscillation between autonomy and synchronization. Sustainable architectural evolution requires calibrated governance density, infrastructural embedding of shared constraints, and disciplined pacing of structural change. These insights can guide enterprise architects in designing

modular, scalable systems that maintain coherence under distributed team dynamics.

REFERENCES

Abd-Elwahab, A. M., Mohamed, A. G., & Shaaban, E. M. (2023). MicroServices-driven enterprise architecture model for infrastructure optimization. *Future Business Journal, 9*, 90. https://doi.org/10.1186/s43093-023-00268-3

Alzoubi, Y. I., & Mishra, A. (2024). Enterprise architecture contribution in distributed agile software development. *Systems Engineering, 27*, 570–584. https://doi.org/10.1002/sys.21739

Berntzen, M., Stray, V., Moe, N. B., & others. (2023). Responding to change over time: A longitudinal case study on changes in coordination mechanisms in large-scale agile. *Empirical Software Engineering, 28*, 114. https://doi.org/10.1007/s10664-023-10349-0

Dursun, H. (2023). Full spec software via platform engineering: Transition from bolting-on to building-in. In *Proceedings of the International Conference on Evaluation and Assessment in Software Engineering (EASE '23)* (pp. 1–4). ACM. https://doi.org/10.1145/3593434.3593440

Fávero, L. F., Almeida, N. R. D., & Affonso, F. J. (2025). A systematic mapping study on the modernization of legacy systems to microservice architecture. *Applied System Innovation, 8*(4), 86. https://doi.org/10.3390/asi8040086

Li, X., d'Aragona, D. A., Cerny, T., & others. (2025). Exploring microservice ownership and organizational coupling in open-source projects: An empirical study. *Computing, 107*, 102. https://doi.org/10.1007/s00607-025-01454-7

Makris, A., Tserpes, K., & Varvarigou, T. (2022). Transition from monolithic to microservice-based applications. Challenges from the developer perspective [version 1; peer review: 2 approved with reservations]. *Open Research Europe, 2*, 24. https://doi.org/10.12688/openreseurope.14505.1

Martínez, D., Castaño, S. L., Perafan, J. C., & Mondragón, O. H. (2026). Microservices-based architecture to support automation applications in smart industries with end-to-end real-time requirements. *Tsinghua Science and Technology, 31*(2), 708–725.

Nguyen-Duc, A., Khanna, D., Le, G. H., Greer, D., Wang, X., Zaina, L. M., Matturro, G., Melegati, J., Guerra, E., Kettunen, P., Hyrynsalmi, S., Edison, H., Sales, A., Chanin, R., Rutitis, D., Kemell, K.-K., Aldaeej, A., Mikkonen, T., Garbajosa, J., & Abrahamsson, P. (2024). Work-from-home impacts on software project: A global study on software development practices and stakeholder perceptions. *Software: Practice and Experience, 54*, 896–926. https://doi.org/10.1002/spe.3306