

Agility vs. Standardisation in Enterprise IT: Managing the DevOps-Enterprise Architecture Paradox in Large Organisations

Palak Dabral

University of Melbourne
palakdabral2002@gmail.com

Yutong Han

University of Melbourne
han261460@gmail.com

Qi Gao

University of Melbourne
qigao1761@gmail.com

Yuechuan Li

University of Melbourne
12631282575@gmail.com

Rod Dilnutt

University of Melbourne
rpd@unimelb.edu.au

Lelia Meratian Esfahani

University of Melbourne
lmeratian@unimelb.edu.au

Abstract

This report critically examines the tensions between DevOps and Enterprise Architecture (EA) in large organisations by focusing on the governance and cultural dynamics that shape their alignment. Drawing on academic literature and real-world cases to show that these tensions are not issues to solve, but challenges to manage. When organisations focus on support instead of control, they achieve better outcomes, including improved alignment and agility. This report also acknowledges current limitations in theory and practice, then recommends decentralising architectural decisions, investing in platform teams, and fostering cross-role cultural alignment. Ultimately, it argues that the EA and DevOps relationship should not be viewed as a structural flaw. By contrast, managing this tension is a strategic capability that enables delivery speed and architectural coherence to coexist in today's complex, fast-changing world.

1. Introduction

The emergence of DevOps in the digital age has stemmed from the increased need for agility in development and continuous deployment in fast-paced and evolving business landscapes. DevOps is a process that emphasises the relationship between 'development' and 'operations', trying to reduce the time it takes to develop and deploy so that internal and external customers get quality products quicker. However, in a Harvard Business Review report (2019) a glaring gap is visible: while 86% of business respondents specified that rapid deployment of software was essential to organisational success, only around 10% of respondents stated that they had accomplished this.

Meanwhile, large organisations have long relied on enterprise architecture (EA) governance frameworks, for instance TOGAF®. It is used to align IT strategy with the business for a long period, as well as provide standardised controls to IT assets (Van Haren, 2011). This form of governance emphasises comprehensive planning and control in terms of enterprise architecture to safeguard strategic consistency. So, when adopting DevOps which promotes the notion of autonomy and iterations, friction with the stability- and standardisation-focused EA models is almost inevitable.

This makes it especially valuable to investigate how large organisations grapple with balancing agility and architecture, and how they can potentially reconcile short-term focus on delivery and long-term strategic alignment without introducing unnecessary loss in product quality.

This report will focus on the following research question: *What tensions arise between DevOps-driven agility and EA standardisation in large organisations with comprehensive IT environments, and how can these be addressed to balance delivery speed with strategic alignment?*

To explore this question, this paper will include a literature review which will outline existing areas of consensus and divergence in current research. Then a discussion section is presented which will critically engage with the reviewed literature and explore potential integration paths and solutions. In doing so, this paper aims to provide both theoretical insights and practical implications for organisations seeking to reconcile agility with strategic coherence during digital transformation.

2. Literature Review

2.1. Key Concepts for EA and DevOps

To understand the tension between DevOps and Enterprise Architecture (EA), we need to clarify what they represent for large organisations, how their priorities are defined, and why those priorities are often at odds with another.

DevOps embodies speed and flexibility. DevOps includes more than just a set of tools or practices; it is a mindset that integrates development and operations teams to deliver software quicker with more reliability. As asserted by Bass et al. (Bass, 2015), this model consists of practices such as continuous integration, real-time feedback, and so forth, which enable rapid iteration. DevOps allows teams to respond quickly, make decisions quickly, and iterate based on real-time feedback, which are all attributes of agility.

By contrast, enterprise architecture is concerned with stability and long-term alignment. Ross et al. (2006) define EA to align IT infrastructure with a broader business strategy. It achieves this by enforcing standardisation, through shared data models, common platforms, controlled processes, and phased planning.

2.2. Agility vs. Standardised

The primary conflict between agility and standardisation arises from their opposing methods for tracking organisational complexity. Although EA requires standardisation through the establishment of shared data models and controlled phased planning to provide long-term organisational stability, this specific type of structural rigidity may unintentionally reduce the possibility for local innovation and ability to respond to immediate business requirements.

Organisations that operate on a very large scale face the challenge of "agile silos", where groups of developers work separately from one another on pushing projects to completion quickly without any horizontal EA oversight (and in many instances with no architectural integrity). This results in technically fast, but architecturally fragmented results as noted by Khan et al. (2022), who point out that communication problems between developers and IT operations represent a major barrier in these situations, as the inability to have a common governance language leads to long delays in project completion. Therefore, the challenge is not just about whether to emphasise speed or order, but how to maintain architectural integrity without introducing bureaucratic inertia.

2.3. Enterprise Architecture Frameworks: Focus on TOGAF®

2.3.1 Focus on Enterprise Architecture-TOGAF®

TOGAF® (The Open Group Architecture Framework®) is one of the most extensively utilised enterprise architecture frameworks globally, developed by The Open Group®.

It provides a standardised methodology known as the Architecture Development Method (ADM) to guide organisations for designing, evolving and managing Enterprise Architectures within the four main domains of Business, Data, Application and Technology.

The objective of TOGAF® is to establish a standard language and structured process, in order to ensure that all IT investment and infrastructure within an organisation are aligned to the overall business strategy of the organisation.

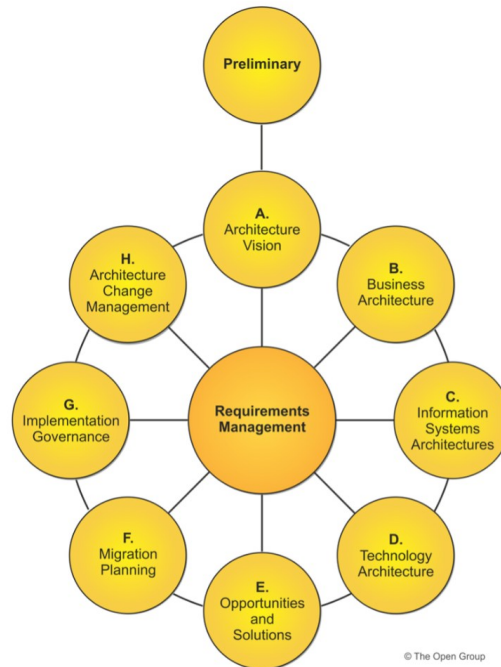


Figure 1. TOGAF® ADM *Opengroup.org, n.d., <https://www.opengroup.org/togaf>.*

The decision to anchor this analysis in The Open Group® Architecture Framework (TOGAF®) is its established position as the worldwide standard for enterprise governance. Rather than being an arbitrary, localised selection, TOGAF® represents a consensus regarding the management of the architectural lifecycle across industries via its authoritative Architecture Development Method (ADM) (Van Haren, 2011). In high-stakes environments such as public sector transitions and complex financial industries (Yudhistira & Fajar, 2024), TOGAF® has evolved into a de facto reference model to help align the evolution of technology with institutional stability over time. Because TOGAF® is near-universally adopted, the friction between TOGAF's® phased, structured methodology and DevOps' iterative, rapid velocity methodology is not only seen as a technical issue but a significant strategic issue faced by large-scale enterprises today (Kotusev, 2018). The pervasive use of TOGAF® in academic literature is also well-documented. As attested by Chanchari et al. (2022), TOGAF® is the asset baseline from which all current "Agile EA" models have been built, thereby providing this research a validated, industry agnostic foundation on which to investigate how to reconcile delivery speed and architectural integrity.

2.3.2 The contradictory relationship between TOGAF® and DevOps

TOGAF® offers a structured governance model or framework for enterprise architecture with focus on standardisation, consistency and the traceability of architectural decisions. While well-matched for delivering long-term alignment, TOGAF® can come across as too structured, formal and slow in situations

where DevOps operates with a goal of shorter feedback cycles and faster continuous delivery (Kotusev, 2018).

The distinction is largely in the governance model. TOGAF® clearly relies on formally appointed enterprise architects and centralised decision making. DevOps encourages autonomy, collaboration, and technical flexibility. DevOps practices commonly use technologies like microservices and containers to satisfy scalability and speed requirements but can create situations where established architectural standards are ignored, resulting in shadow IT and architectural drift, which creates a strategic disconnect that undermines organisational coherence.

These conflicts come to the fore in practice frequently. For example, Uludağ et al. (2022) researched five German businesses and found that there was common confusion between EA architects and DevOps delivery teams on goals and responsibilities. The similar tensions in the financial services sector where EA teams often get stuck with EA governance while delivery DevOps type teams dance through to evolve and release capabilities frequently, leading to strategic disconnects and organisational conflicts.

However, this is not to imply that EA frameworks like TOGAF® cannot work in an agile way. Some researchers have tried to build models to connect TOGAF® with an agile way of working.

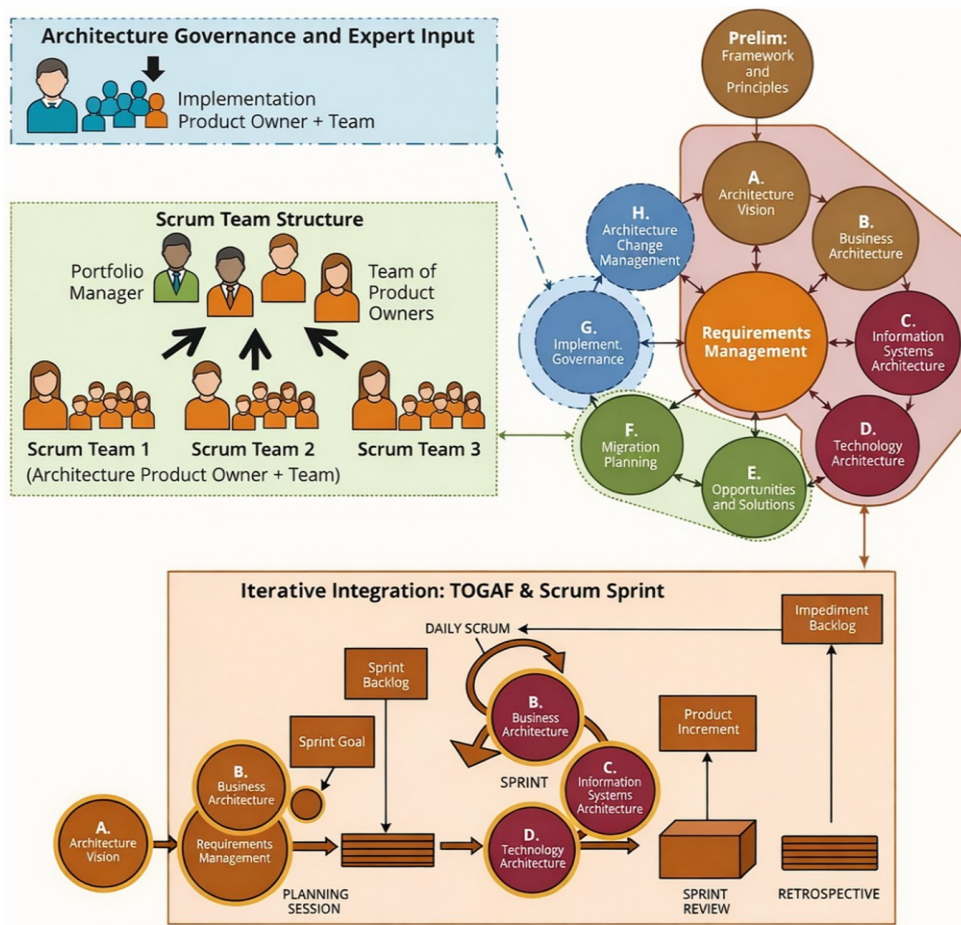


Figure 2. Integrating TOGAF® ADM with Agile Scrum (Hanschke et al., 2015)

For example, Hanschke et al (2015) put forward a hybrid FOR agile approach, with the TOGAF® ADM, treating the ADM phases as agile project iterations. In this hybrid model they also formed an Architecture

Product Backlog and incrementally delivered products to support agility and keep the structure of EA in place.

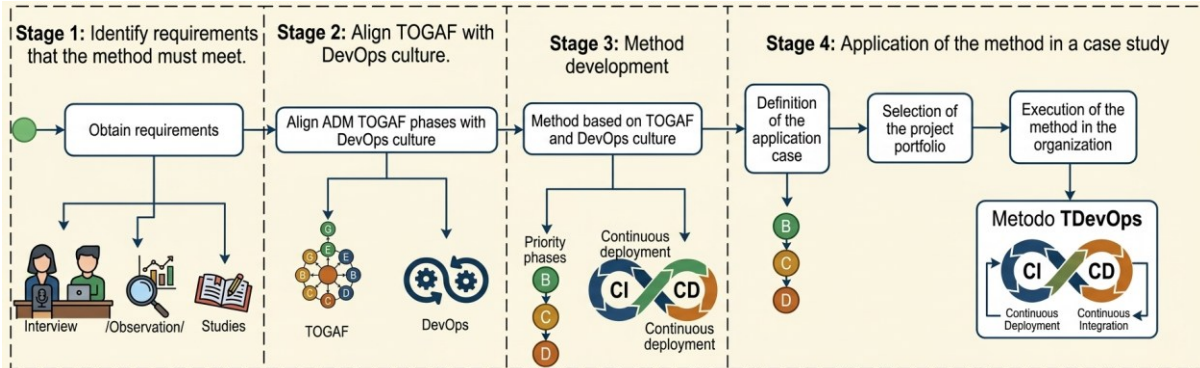


Figure 3. Proposed Method Based on TOGAF® and DevOps (Chanchari et al., 2022).

At the application level, Chanchari et al. researched 'TDevOps' methodology at an application level in 2022, which is a four-stage approach for aligning TOGAF® Phase B (Business) Phase C (Information Systems) and Phase D (Technology Architecture).

Unlike purely theoretical models, this research integrated the four stages of TDevOps with the collaborative aspects of DevOps culture. TDevOps was validated using three separate large scale software projects that focused on aligning stakeholder requirements to Continuous Integration and Continuous Deployment (CI/CD) methodologies. These empirical findings indicated significant improvement in operational indicators, such as reduced delivery lead times and decreased frequency of released errors.

These cases and frameworks suggest that while DevOps and EA operate on different principles, they are not mutually exclusive. Instead, they represent complementary forces that, when integrated thoughtfully, can drive innovation without sacrificing architectural coherence.

2.4. DevOps and Governance

2.4.1 DevOps Is More Than a Toolset

Existing research generally agrees that DevOps cannot be understood simply as a set of automation tools. More accurately, it is an organisational capability built around continuous delivery, cross-functional collaboration, and rapid feedback. Leite et al. (2019) pointed out that DevOps is essentially a collaborative effort across the development and operations boundaries, aiming to improve delivery speed while maintaining system reliability. Research by Erich et al. (2017) and Lwakatаре et al. (2019) also shows that the effectiveness of DevOps is usually not due to a single tool, but rather occurs in conjunction with changes in team collaboration methods, role adjustments, automated pipelines, and deployment responsibilities.

This is crucial to the present study. The impact of DevOps on Enterprise Architecture (EA) is not simply about software being released faster. A deeper change lies in the reorganisation of delivery activities themselves. A systematic review by Shahin et al. (2017) shows that continuous integration, continuous delivery, and continuous deployment shorten feedback cycles and change how issues are discovered, addressed, and released. Lwakatаре et al. (2019) further point out that as deployment responsibility shifts to development teams, some decisions that were previously made at phase boundaries enter daily development activities. This makes governance methods relying on sparse approval points less timely and less effective.

Therefore, the challenges brought by DevOps are not merely about speed. More importantly, it changes when governance can intervene and what it must respond to.

2.4.2 Autonomy and Governance Boundaries

DevOps is often associated with team autonomy, but this does not mean that governance can be removed. Boh and Yellin (2006) point out that the important role of enterprise architecture standards is to provide shared principles and common constraints for IT resource management, thereby reducing the disruption of overall coordination caused by local decisions. Van der Raadt and van Vliet (2008) also emphasise that the EA function is not just architectural documents or technical artifacts, but also includes a whole set of activities and arrangements surrounding decision-making, compliance, and organisational collaboration.

From this perspective, the problem facing large organisations is not whether governance is still needed, but how governance should be integrated into team-level decision-making. Without shared platforms, interface constraints, data standards, and clear escalation paths, team autonomy can easily slip into local optima, thereby compromising enterprise-wide architectural consistency.

Empirical research supports this view. Salameh and Bass (2022), in a long-term case study within a multinational fintech organisation, found that delegating some architectural decisions closer to delivery, while retaining Architecture Owners, lateral knowledge sharing, and architectural change management processes, helps improve architectural alignment without compromising team autonomy. In other words, DevOps does not replace governance with autonomy, but rather requires governance to shift from external control to boundary-based, distributed, and continuous coordination.

2.4.3 From Ex Ante Control to Continuous Control

In a continuous delivery environment, both the objects and forms of governance are changing. Traditional architectural governance typically relies on early design, phase reviews, and compliance checks. In contrast, DevOps places greater emphasis on integrating testing, deployment, monitoring, and feedback into a continuous process. Shahin et al. (2017) pointed out that the core of continuous practices is not just increasing release frequency, but embedding quality assurance and change control into the daily delivery process.

The study by Plant et al. (2022) further demonstrates that in a DevOps environment, organisations often do not completely abandon traditional controls, but rather form a hybrid model where manual and automated controls coexist. This combination is significantly influenced by risk appetite and DevOps maturity. Therefore, DevOps has not eliminated governance, but rather has driven a gradual shift in governance from ex-ante approval to continuous control.

From an architectural perspective, this shift is crucial. Architectural efforts no longer rely primarily on documentation and periodic approvals to maintain influence, but increasingly depend on principles, interface constraints, automated checks, change processes, and runtime feedback. For the present study, this means that as more and more technical decisions are pushed to teams and pipelines, the effectiveness of EA governance will gradually decline if they still primarily rely on proactive, centralised governance logic.

2.5. Alignment of Governance and Strategy

2.5.1 The Continuing Role of Enterprise Architecture

In large organisations, the value of enterprise architecture lies not in slowing down delivery, but in maintaining consistency across teams, systems, and timescales. Ross et al. (2006) view EA as the organisational foundation supporting business execution, emphasising its relationship with process integration, standardisation, and execution capabilities. Boh and Yellin (2006) also point out that EA standards help organisations manage IT resources more effectively, thereby improving overall coordination. Combined with the discussion of van der Raadt and van Vliet (2008), it can be seen that the role of EA is not merely to provide a blueprint, but to establish a set of constraints and coordination mechanisms for enterprise-level decision-making that can be organised and executed.

The problem is that traditional EA governance often relies on periodic reviews, centralised approvals, and document compliance checks, which are not entirely compatible with the continuous pace of DevOps. Plant et al. (2022) showed that in a DevOps environment, decentralised decision-making and high levels of automation can create practical difficulties for auditing, control, and verification. However, they also pointed out that organisations typically do not move towards complete de-governance, but rather recombine manual and automated controls based on risk appetite and maturity.

This demonstrates that strategic alignment has not lost its importance in a DevOps context. The real change is that alignment no longer relies primarily on centralised approval, but rather on principle setting, risk boundaries, exception escalation mechanisms, and continuous control.

2.5.2 Conditions for Distributed Governance

Existing research is increasingly demonstrating that distributed governance is not simply about decentralisation. The case study by Salameh and Bass (2022) shows that a more stable balance between team autonomy and architectural consistency can only be achieved when organisational structure, role assignments, and architectural change management processes are adjusted simultaneously. The key is not whether a central architectural role completely withdraws, but how architectural work is reorganised to a position closer to delivery, while still maintaining coordination mechanisms for cross-team, high-impact issues.

In this sense, distributed governance requires at least three conditions. First, decision-making boundaries must be clear; the team needs to know which problems can be resolved locally and which will affect shared assets and enterprise-level risks. Second, coordinating roles must be clearly defined; otherwise, cross-team dependencies and architectural disagreements will accumulate. Third, there must be effective escalation paths to ensure that high-risk issues can be identified and addressed promptly.

The system mapping study by Fuentes-Quijada et al. (2024) also supports this judgment. They point out that research related to BizDevOps, EA, and IT governance is increasingly focused on how to achieve business-IT alignment in a faster delivery environment, but existing literature is still mainly advisory, with relatively limited solid empirical evidence. Therefore, a more reasonable approach at present is not to presuppose a certain governance model as the standard answer, but to analyse how different governance arrangements coordinate speed, consistency, and risk control under specific organisational conditions.

2.6. Gaps in the Literature

Existing research has sufficiently demonstrated the existence of tension between DevOps and EA/IT governance, but explanations of how this tension is specifically formed, allocated, and managed within organisations remain insufficient.

First, explanations at the level of mechanisms are still inadequate. DevOps research has clearly described the main practices of continuous delivery, automation, and cross-functional collaboration, and also pointed out that these practices alter the pace of feedback and the granularity of change (Shahin et al., 2017). Meanwhile, governance research has begun to discuss how control is shifting from manual review to automated and hybrid control (Plant et al., 2022). However, existing literature still lacks a systematic explanation of how continuous delivery changes the timing of intervention, form of evidence, exception handling, and allocation of responsibility in architectural governance. In other words, the literature can explain why tension exists, but it doesn't explain in enough detail how this tension is governed.

Second, research on the boundaries of decision-making authority remains insufficient. EA research has long emphasised the importance of shared standards and enterprise-level constraints (Boh & Yellin, 2006), while research in the DevOps context suggests that some architectural decisions need to be closer to team and delivery activities (Salameh & Bass, 2022). However, a comparable and transferable analytical framework is still lacking between these two types of literature to explain which decisions are suitable to remain at the

team level, which decisions must remain at the enterprise level, and how these boundaries adjust with changes in organisational size, coupling, and risk exposure.

Third, the existing evidence base remains weak. Fuentes-Quijada et al. (2024) explicitly pointed out that there are many recommendation-oriented studies in this field, but relatively few verifiable empirical studies. While existing valuable research has provided important insights, much of it comes from a limited number of organisations and specific case scenarios, such as single cases, small-scale cases, or embedded studies in specific industries (Erich et al., 2017; Lwakatere et al., 2019; Salameh & Bass, 2022). This means that current literature is better at proposing "potentially effective coordination methods" than at proving whether these methods can be stably replicated across different organisations.

Finally, the organisational logic behind the transformation of architect roles remains poorly explained. Van der Raadt and van Vliet (2008) pointed out that the EA function essentially encompasses organisational capabilities and role assignments, not just architectural artifacts themselves. More recent research also shows that in more agile and continuous delivery environments, architectural work is indeed shifting closer to team and operational levels (Salameh & Bass, 2022). However, a key question remains unanswered: how do architects transition from gatekeepers to enablers or coordinators, gaining new organisational legitimacy in the process? Existing research has indicated the direction of this role migration, but lacks cross-contextual, long-term explanations on the conditions under which this transformation can be sustained, when it might fail, and how it, in turn, impacts governance performance.

Based on these shortcomings, this article does not view the tension between EA and DevOps as a structural defect waiting to be eliminated, but rather as an organisational and strategic capability issue that large organisations need to continuously address in their digital transformation.

3. Discussion

3.1. Governance Tension: Centralised vs Decentralised

One of the largest tensions in getting Enterprise Architecture (EA) into DevOps alignment is governance, specifically the struggle between centralised control and decentralised freedom (Fuentes-Quijada et al., 2024). Rigid centralisation slows down delivery, but pure decentralised approaches can cause fragmentation and strategic drift. The problem is how to create a dynamic balance that maintains both speed and coherence, where governance guides without commanding.

Using institutional theory, we can see this tension as the result of competing institutional logics (Smith & Tracey, 2016). EA governance procedures are typically shaped by coercive forces that tend towards formality and standardisation (Chen et al., 2024). But DevOps originates from mimetic forces and normative forces based on agile and lean principles (Karunaratne et al., 2024). These logics coexist in the same organisation but with varying priorities; therefore, conflict exists on how to implement governance (Peerzada, 2025).

Traditional EA governance was designed around formal review cycles and phased decision-making, which assumes there's time to examine architectural implications before commitments are made (Boh & Yellin, 2006). DevOps decisions get made fast, in the middle of a sprint, by people who are focused on shipping. By the time a traditional architecture review could intervene, the decision has often already been implemented (Shahin et al., 2017).

In practical environments, organisations have approached this differently. ING's tribes and squads model is a great example of how decentralised delivery can coexist with centralised guidance (Calnan & Rozen, 2019). Rather than telling architecture top-down, ING uses architectural principles, alignment boards and shared services to maintain coherence. Teams can then make local decisions within clear strategic boundaries. This meant governance was present without being a bottleneck. Spotify, by contrast, took a decentralised approach, relying heavily on team autonomy and internal platforms. While this drove initial

innovation, ultimately, they faced fragmentation, duplicated effort and tool sprawl. Spotify's experience is picked up in the following section, as its problems were as much cultural as structural.

What ING's model demonstrates is that effective governance doesn't have to mean control. When architectural principles are embedded into how teams operate day-to-day, rather than applied as external checkpoints, delivery speed and strategic coherence can coexist.

3.2. Cultural and Structural Friction

While governance is often seen as a structural problem, cultural friction is just as important and often harder to fix. EA and DevOps operate under fundamentally different assumptions, where EA values stability, foresight and planning, while DevOps prioritises experimentation, feedback and responsiveness. Even when structural models are well-designed, these divergent mindsets create misunderstandings that erode collaboration.

EA and DevOps people tend to think differently, and this shows up in everyday situations. Architects often get frustrated watching DevOps teams make technology choices without considering long-term implications. Engineers, on the other hand, frequently see architects as people who slow things down and don't understand delivery pressures (Uludağ et al. 2022). These aren't just personality clashes, it reflect genuinely different professional values that have been reinforced over years of working in different ways. Even when governance structures allow for flexibility, cultural mismatches can still erode trust and collaboration.

Spotify is a good example of what happens when this gets ignored. Their early transformation gave teams enormous autonomy, which worked well at first (Bäcklander, 2019). Innovation was fast, teams were motivated, and the model got a lot of attention. But over time the cracks showed. Teams started making inconsistent decisions, tools proliferated, and practices diverged across the organisation. The problem was that nobody had done the work of building shared norms and a common understanding of what good looked like. When everyone is free to do things their own way, you eventually end up with an organisation that can't coordinate effectively. Spotify had to bring in platform teams and shared governance later, but by that point it was remedial work rather than proactive design.

ING handled this differently, though it wasn't easy either. When they restructured into tribes and squads, they were explicit about the cultural shift they were asking for. Architects had to genuinely change how they operated and be more advising and collaborating (McKinsey & Company, 2017). Engineers had to engage with architectural thinking rather than treating it as someone else's problem. That kind of change takes time and doesn't happen just because leadership announces it. What made it work at ING was sustained investment in cross-role relationships and a genuine effort to align how different teams understood their shared goals.

The broader point is that structural reforms set the conditions for cultural change, but they don't guarantee it. If the people involved don't trust each other or don't understand each other's pressures, even a well-designed governance model will struggle. This is why the EA-DevOps tension can't be treated purely as an organisational design problem; it is the human side of it matters just as much.

3.3. Practical Strategies: Embedding and Adaptation

So how are organisations dealing with this in practice? The honest answer is that the ones doing it well aren't trying to resolve the tension between EA and DevOps, they're learning to work within it. That means bringing architecture closer to where decisions actually happen, rather than keeping it as a separate oversight function that teams have to report to (Salameh & Bass, 2022).

One of the most common strategies is embedding architects directly into product or feature teams. Instead of reviewing solutions from afar, architects participate early in the design process, helping teams make informed trade-offs as they go (PMI Disciplined Agile, 2018). This role shifts from enforcer to collaborator,

which can make governance feel less like a burden and more like a support function. But it only works if the architect is trusted and if the team values architectural thinking. In high-performing teams, this setup allows governance to scale naturally with delivery (Nicolaidis, 2019).

ING's model reflects this idea on a broader scale. Their squads are autonomous but aligned through architectural boards and shared services, allowing teams to move quickly while remaining within enterprise boundaries (Gavandi, 2021). These mechanisms provide structure without micromanagement. While ING built architectural guidance into its delivery model from the beginning, Spotify's strategy was more reactive, addressing alignment only after delivery friction emerged. Compared to Spotify's earlier approach, ING's model offers more built-in safeguards (Kolk, 2015). Spotify responded to these issues by introducing platform teams to centralised infrastructure groups, which provided reusable services and architectural support (Charron, 2013). This helped re-establish architectural alignment but required a shift in both approach and operating model (ThoughtWorks, 2020).

Together, these strategies show that most successful large organisations tend to reduce the weight of centralised governance, bring architectural thinking closer to where decisions are made, and ensure flexibility without losing vision of enterprise goals (O'Rourke et al., 2023). This diversity of approaches shows that the answer to our research question is a flexible governance perspective tailored to organisational context. Different organisations will land in different places depending on their size, culture, and how far along their DevOps journey they are. What they share is a move away from governance as gatekeeping and toward governance as something that travels with the team rather than sitting above it.

4. Implication, Recommendations and Limitations

4.1. Implications

For practitioners, these findings suggest that resolving DevOps-EA tensions requires both structural change and mindset shift. Architects must engage at the team level, while engineering leads must appreciate the long-term impact of architectural decisions. Moreover, adopting platform thinking and internal governance mechanisms can decentralise architectural decision-making while maintaining long-term coherence and risk management. Evidence from platform-based ventures shows that distributed governance, supported by transparent team practices, enables effective decision-making without undermining overall strategic objectives (Martino et al., 2024).

For researchers, this study extends existing work on competing institutional logic into the DevOps-EA context. Future empirical research should investigate how these tensions manifest across industries, especially in highly regulated sectors or those with legacy constraints.

4.2. Recommendations

R.1 Establish Principle-Based Governance

Organisations should move away from rigid stage-gate models and adopt lightweight, principle-driven architectural governance. Techniques such as architecture fitness functions (Much, 2025) and guardrails promote compliance without stifling team autonomy. For instance, Amazon® has implemented pre-approved blueprints as guardrails, providing standardised reference architectures that guide teams while allowing safe innovation. Complementing this, automated compliance checks in cloud environments act as fitness functions, continuously validating that deployments adhere to enterprise standards such as encryption, security policies, and architectural principles (Markov, 2025). Together, these mechanisms empower teams to experiment and iterate rapidly without compromising core requirements, demonstrating how principle-based governance can operate effectively in practice.

R.2 Embed Enterprise Architects in Delivery Teams

Following models seen at companies like Capital One, enterprise architects should shift from enforcing standards to enabling decisions, working alongside teams to guide architectural evolution iteratively. At

Capital One, this shift allowed architects to act as “technology coaches,” improving collaboration between teams and reducing bottlenecks traditionally associated with centralised approvals. As a result, teams were able to deliver changes more rapidly while still aligning with enterprise-wide architectural goals.

R.3 Invest in Platform Teams and Internal Developer Portals

Investing in platform teams and developer portals enables standardisation through reusable tools and infrastructure, supporting team autonomy while maintaining architectural consistency and reducing friction between DevOps and enterprise architecture (Chandrasekaran, 2024). This model is evident in organisations like Spotify, where platform teams provide reusable infrastructure and developer tools that standardise practices across squads while preserving autonomy. The introduction of these internal platforms helped reduce duplication, improve deployment consistency, and re-establish architectural coherence after earlier fragmentation challenges (Charron, 2013; ThoughtWorks, 2020).

R.4 Drive Cross-Role Cultural Alignment

DevOps and EA communities often operate in silos, each with distinct goals and practices. Promoting shared ownership, encouraging CI/CD practices, and integrating roles across boundaries (Andersen, 2024) can help create a unified culture focused on shared organisational goals. For instance, ING’s agile transformation emphasised cross-functional collaboration and shared accountability between architecture and delivery teams, supported by aligned performance metrics and continuous feedback. This helped reduce resistance between roles and ensured that architectural considerations were integrated into everyday decision-making rather than treated as external constraints (Calnan & Rozen, 2019). Such cases highlight that cultural alignment is not abstract, but a measurable driver of coordination and delivery effectiveness.

4.3. Limitations

The discussion of the alignment of Enterprise Architecture (EA) and DevOps is relatively new and highly fragmented. A lot of current research focuses on DevOps or EA individually, making it hard to find comprehensive studies that examine how the two actually cooperate in practice. This paper was thus compelled to draw on a limited set of examples, which may not reflect the full range of how organisations manage these tensions.

Another limitation is that most of the examples used in the discussion, such as ING and Spotify, are from large, technologically advanced companies operating in adaptable environments. While useful, their strategies may not easily apply to more traditional industries with strict regulations, legacy systems, or limited resources, where prior research suggests that organisational inertia and compliance requirements can significantly constrain agile transformation efforts (Boh & Yellin, 2006; Erich et al., 2017).

Finally, this report is mostly based on a theoretical and interpretive rather than experiential research process like interviews or field observations. For that reason, it doesn't capture the everyday challenges and lived experiences of teams trying to bridge the gap between EA and DevOps in real time. In the future, clearly more grounded research is needed. Follow-up research can then explore the difference between industry and organisational size to see how these tensions play out in varying contexts. Qualitative studies like case studies, interviews, or even observational work can be a means of shedding light on the pragmatic realities that teams face.

5. Conclusion

This paper sets out to explore the tensions between Enterprise Architecture (EA) and DevOps in large organisations, focusing on how governance models and cultural dynamics shape their alignment. The findings indicate that while EA and DevOps often appear in conflict, centralised control vs decentralised agility, standardisation vs experimentation, they can complement one another when carefully integrated. Through theoretical framing and industry examples, the discussion highlighted how governance tensions stem from competing institutional logics and how structural reforms alone are insufficient without

corresponding cultural alignment. The embedded architect model, platform governance, and shared architectural principles emerged as key enablers for EA to remain relevant in agile delivery environments. Ultimately, successful alignment does not mean removing the tension but managing it in a way that fosters both coherence and speed. EA needs to shift from being a strict rule-setter into a facilitative, value-driven role that empowers teams without sacrificing strategic direction.

Bibliography

- Andersen, G. (2024). The impact of DevOps culture on software architecture and collaboration. *MoldStud*. <https://moldstud.com/articles/p-the-impact-of-devops-culture-on-software-architecture-and-collaboration>
- Bäcklander, G. (2019). Doing complexity leadership theory: How agile coaches at Spotify practise enabling leadership. *Creativity and Innovation Management*, 28(1), 42–60. <https://doi.org/10.1111/caim.12303>
- Bass, L., Weber, I., & Zhu, L. (2015). *DevOps: A software architect's perspective*. Addison-Wesley Professional.
- Boh, W. F., & Yellin, D. (2006). Using enterprise architecture standards in managing information technology. *Journal of Management Information Systems*, 23(3), 163–207. <https://doi.org/10.2753/MIS0742-1222230307>
- Calnan, M., & Rozen, A. (2019). ING's agile transformation—Teaching an elephant to race. *Journal of Creating Value*, 5(2), 190–209. <https://doi.org/10.1177/2394964319875601>
- Chanchari, M., Huanca Torres, F. A., Asin, F., & Saboya, N. (2022). The method based on the TOGAF framework and DevOps culture in the implementation and deployment of software. In *World Conference on Information Systems and Technologies* (pp. 98–109). Springer. https://doi.org/10.1007/978-3-031-04829-6_10
- Chandrasekaran, S. (2024). Optimizing software quality through internal developer portals. *International Journal of Science and Research*, 13(1), 696–699. <https://doi.org/10.21275/sr24110013829>
- Charron, T. (2013, April 9). Scaling agile at Spotify: An interview with Henrik Kniberg. *InfoQ*. <https://www.infoq.com/news/2013/04/scaling-agile-spotify-kniberg/>
- Chen, Y., Ma, H., & Zhou, T. (2024). Learn from whom? An empirical study of enterprise digital mimetic isomorphism under the institutional environment. *Economies*, 12(9), 243. <https://doi.org/10.3390/economies12090243>
- Competitive advantage through DevOps. (2019). *Harvard Business Review Analytics Services*. <https://hbr.org/resources/pdfs/comm/google/CompetitiveAdvantageThroughDevOps.pdf>
- Erich, F. M. A., Amrit, C., & Daneva, M. (2017). A qualitative study of DevOps usage in practice. *Journal of Software: Evolution and Process*, 29(6), e1885. <https://doi.org/10.1002/smr.1885>
- Fuentes-Quijada, G., Ruiz-González, F., & Caro, A. (2025). Enterprise architecture and IT governance to support the BizDevOps approach: A systematic mapping study. *Information Systems Frontiers*, 27(3), 865–888. <https://doi.org/10.1007/s10796-024-10473-2>
- Gavandi, R. (2021, October 7). Case study: Scaled agile implementation at ING in a microservices culture. *Medium*. <https://medium.com/@roshangavandi/case-study-scaled-agile-implementation-at-ing-in-a-microservices-culture-a5eaea8df72>
- Haag, S., & Eckhardt, A. (2017). Shadow IT. *Business & Information Systems Engineering*, 59, 469–473. <https://doi.org/10.1007/s12599-017-0497-x>

- Hanschke, S., Ernsting, J., & Kuchen, H. (2015). Integrating agile software development and enterprise architecture management. In *Proceedings of the 48th Hawaii International Conference on System Sciences* (pp. 4099–4108). IEEE. <https://doi.org/10.1109/HICSS.2015.494>
- Haren, V. (2011). *TOGAF version 9.1*. Van Haren Publishing.
- Karunarathne, M., Wijayanayake, J., & Prasadika, J. (2024). DevOps adoption in software development organizations: A systematic literature review. <https://doi.org/10.1109/ICARC61713.2024.10499789>
- Khan, M. S., Khan, A. W., Khan, F., Khan, M. A., & Whangbo, T. K. (2022). Critical challenges to adopt DevOps culture in software organizations: A systematic review. *IEEE Access*, *10*, 14339–14349. <https://doi.org/10.1109/ACCESS.2022.3140000>
- Kolk, H. (2015). Architecting for 400 agile squads [Conference presentation]. Agile & Software Architecture Symposium.
- Kotusev, S. (2018). TOGAF-based enterprise architecture practice: An exploratory case study. *Communications of the Association for Information Systems*, *43*(1), Article 20.
- Leite, L., Rocha, C., Kon, F., Milojevic, D., & Meirelles, P. (2019). A survey of DevOps concepts and challenges. *ACM Computing Surveys*, *52*(6), Article 127. <https://doi.org/10.1145/3359981>
- Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., Kuvaja, P., Mikkonen, T., Oivo, M., & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*, 217–230. <https://doi.org/10.1016/j.infsof.2019.06.010>
- Markov, R. (2025, April 21). Empower your teams with modern architecture governance. *AWS Architecture Blog*. <https://aws.amazon.com/blogs/architecture/empower-your-teams-with-modern-architecture-governance/>
- Martino, P., Vanacker, T., Filatotchev, I., & Bellavitis, C. (2024). (De)centralized governance and the value of platform-based new ventures. *Small Business Economics*, *64*(4), 1763–1790. <https://doi.org/10.1007/s11187-024-00964-6>
- McKinsey & Company. (2017). ING's agile transformation. <https://www.mckinsey.com/industries/financial-services/our-insights/ings-agile-transformation>
- Much, T. (2025). Fitness functions for your architecture. *InfoQ*. <https://www.infoq.com/articles/fitness-functions-architecture/>
- Nicolaides, C. (2019). What enterprise architecture looks like in a fast-paced tech company. *Capital One*. <https://www.capitalone.com/tech/software-engineering/what-enterprise-architecture-looks-like-at-capital-one/>
- O'Rourke, B., Schroeder, C., Schroeder, S., Vavoulioti, M., & Williams, P. (2023). A modern enterprise architecture is essential for scaling agile. *Bain & Company*. <https://www.bain.com/insights/a-modern-enterprise-architecture-is-essential-for-scaling-agile/>
- Peerzada, I. (2025). Agile governance: Examining the impact of DevOps on PMO: A systematic review. <https://doi.org/10.20944/preprints202508.2184.v1>
- Plant, O. H., van Hillegersberg, J., & Aldea, A. (2022). Rethinking IT governance: Designing a framework for mitigating risk and fostering internal control in a DevOps environment. *International Journal of Accounting Information Systems*, *45*, 100560. <https://doi.org/10.1016/j.accinf.2022.100560>
- PMI Disciplined Agile. (2018). Supporting agile team governance. <https://www.pmi.org/disciplined-agile>
- Ross, J. W., Weill, P., & Robertson, D. C. (2006). *Enterprise architecture as strategy: Creating a foundation for business execution*. Harvard Business Review Press.

- Salameh, A., & Bass, J. M. (2022). An architecture governance approach for agile development by tailoring the Spotify model. *AI & Society*, 37(2), 761–780. <https://doi.org/10.1007/s00146-021-01240-x>
- Senapathi, M., Buchan, J., & Osman, H. (2018). DevOps capabilities, practices, and challenges: Insight from a case study. In *EASE 2018*. <https://doi.org/10.1145/3210459.3210465>
- Shahin, M., Ali Babar, M., & Zhu, L. (2017). Continuous integration, delivery and deployment. *IEEE Access*, 5, 3909–3943. <https://doi.org/10.1109/ACCESS.2017.2685629>
- Smith, W. K., & Tracey, P. (2016). Institutional complexity and paradox theory. *Strategic Organization*, 14(4), 455–466. <https://doi.org/10.1177/1476127016638565>
- ThoughtWorks. (2025). *Technology Radar Vol. 23*. <https://www.thoughtworks.com/radar>
- Uludağ, Ö., Reiter, N., & Matthes, F. (2020). Improving the collaboration between enterprise architects and agile teams. *Intelligent Systems Reference Library*, 347–366. https://doi.org/10.1007/978-3-030-49640-1_18
- van der Raadt, B., & van Vliet, H. (2008). Designing the enterprise architecture function. In *Quality of software architectures* (pp. 103–118). Springer. https://doi.org/10.1007/978-3-540-87879-7_7
- Yudhistira, A., & Fajar, A. N. (2024). Integrating TOGAF and big data for digital transformation. *Sinkron: Jurnal dan Penelitian Teknik Informatika*, 8(2), 1215–1225.