Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 7s (2025)

P&C insurance Application Modernization to SAAS based cloud platforms

Kiran babu Boddapati
Acharya Nagarjuna University, Andhra Pradesh, India
shivakiran02@gmail.com

Article History:
Received: 21-06-2025
Revised: 22-07-2025
Accepted: 20-08-2025

Abstract:

P&C insurers are transforming past policy, billing and claims ecosystems to become more agile, resilient and regulation-ready in order to keep up with the growing digital customer demands. One of the dominant directions is the shift to the use of the SaaS-based cloud platforms, according to which the core insurance services are offered in the form of configurable services and expanded with the help of cloud-native integration and domain services. Modernization programs in insurance systems are often ineffective because of their disjointed migration implementation, lack of quality engineering in the transition process, and discovery of security, data, and operational constraints late in the process. The given paper introduces a balanced modernization blueprint on P&C application portfolios moving to SaaS-based cloud environments. The blueprint proposed will incorporate: (1) a capability-based scoping model which categorizes insurance capabilities between commodity and differentiating workloads, (2) a SaaS-based reference architecture which integrates domain services, event-driven integration, and data governance, (3) a phased migration implementation plan with clear quality gates to reduce the risk of post-migration reliability and maintainability, (4) a DevSecOps-based integrated quality engineering strategy to generate continuous test, security, and audit evidence. Governance mechanisms regarding vendor and lock-in risk management, compliance mapping, and production operating model readiness are also described in the paper with the point that to modernize insurers, operational discipline is just as important as architectural transformation. Lastly, it talks about constraints and research directions in the future, such as comparative empirical assessment over insurers, quantitative migration risk scoring, and refinement of multi-tenancy and observability scheme with catastrophe scale loads.

Keywords: Capability-Driven Scoping; DevSecOps-Integrated Quality Engineering; Event-Driven Integration; Governance and Compliance Management; SaaS-Based Core System Migration; Vendor Lock-In Risk Management.

1. Introduction

Property and Casualty (P&C) insurers are being placed on a long-lasting heat to modernize primary policy administration, billing, claims, and customer engagement functions as well as enhance speed-to-market, resilience, and regulatory stance. The legacy P&C stacks, characterized by strongly coupled monoliths, batch-based integration, environment-specific releases, find it challenging to enable continuous product innovation (e.g. usage based insurance, real time fraud indicators, parametric add-ons) and omni-channel servicing needs. The operating models, based on cloud platforms and Software-as-a-Service (SaaS) have thus emerged as one of the main modernization channels not only in infrastructure cost

rationalization but also in the architectural flexibility and agility, operational scalability, and ecosystem interoperability [1], [2].

Nonetheless, modernization is not a lift and shift venture. Empirical and review-based studies indicate that cloud migration frameworks often focus inadequately on quality aspects, especially maintainability, testability, observability and post-migration reliability, even though these elements are some of the key motivators of long-term modernization success [5]. Likewise, the shift of monoliths to service-oriented or microservice-based decompositions comes with new risk of distributed-system (e.g., of data consistency, failure modes, latency amplification) in addition to augmented deployability and independent scalability [6], [7]. Insurers are faced with these technical risks (enhanced by domain requirements): High auditability, a data retention and data residency requirement, third-party risk management, and stringent security controls of personally identifiable information (PII) and financial transactions [10], [14].

Research on SaaS adoption has clarified that the factors behind the adoption differ by the type of application and the contexts of an organization, and some of those factors are uncertainty, strategic value, and social influence in enterprise adoption decisions [3]. These dynamics of adoption are observed in P&C modernization whereby (a) SaaS core platform choices, (b) cloud-native rebuilds, (c) staged strangler migrations, and (d) hybrid models are choosing to relocate commodity functions to SaaS and continue to differentiate capabilities through custom-built means. The evidence on cloud migration also suggests that organizations should provide systematic characterization of migration strategies and limitations in order to prevent discontinuous implementation and tooling breaks [4]. Also the vendor lock-in and interoperability risk should be addressed using portability strategies, decoupling of architecture and contract conscious escape mechanisms [12], [20].

The given paper, called P&C Insurance Application Modernization to SaaS-based Cloud Platforms, suggests a systematic process of modernization that would combine (1) a domain-sensitive reference architecture of SaaS-based P&C capabilities, (2) a progressive model of migration implementation in line with continuous delivery and DevOps, and (3) quality and security-by-design engineering strategy that would be informed by cloud migration, microservice, data quality and cloud security literature sources [8]-[11]. It aims to deliver a publication research-supported roadmap that can be modified by the insurers and platform teams to make a safe upgrade, maintain the continuity of business, compliance, and customer trust.

2. Background

The common definition of cloud computing is that it allows the on-demand access to common computing resources and elastic scaling of computing, which reorients procurement and operations on capital-intensive infrastructure to consumption of services [2]. On the business side, the reasons of cloud adoption are agility, scalability and cost-efficiency, whereas it also raises the questions of governance, including risk, sourcing strategy and organizational change

[1]. In the case of P&C insurers, these trade-offs are compounded since core platforms have become entangled with regulatory and actuarial processes and that modernization needs to avoid breaking the integrity of pricing, coverage interpretation, claims adjudication and financial controls.

The studies in SaaS adoption offer a viable background to the choices of modernization. Benlian, Hess, and Buxmann demonstrate that adoption of SaaS is influenced by both driving and restraining factors as well as the application type where enterprise adoption is not entirely a technical decision but a socio-technical and economic one [3]. This difference is more evident in insurance where commodity modules (CRM, HR, collaboration, some analytics tooling) are rapidly shifting to SaaS, and core policy/claims systems have more obstacles owing to integration complexity, the sensitivity of data, and customization requirements.

There is also interaction of modernization and architectural development studies. Methodical surveys of cloud migration discover the necessity of classification structures to encompass migration maturity, limitations, tool support and architectural adaptation needs [4]. In recent literature reviews it is emphasized that a high number of legacy-to-cloud solutions do not have the robust end-to-end quality factor considered, which subsequently adds to the post-migration quality challenge and operational friction [5]. In the meantime, studies of microservices also reflect the advantages as well as risks of decomposition such as better independent deployability and scalability on the one hand and augmented coordination, distributed tracing requirements, and transaction/data difficulties on the other hand [6], [7]. Recurrent migration studies validate such problems as service naming, team process shift, and toolchain operational maturity, which is frequently associated with success in DevOps potential and automation [6], [8].

Regarding delivery, DevOps and continuous delivery literature focuses on the reduction of the feedback loops and the combination of development and operations to enhance release reliability and speed [8], [9]. There are tremendous benefits of continuous delivery, but it also causes organizational changes, the level of automation, and the attentive management of the non-functional requirements within the pipeline [9]. Delivery maturity is no exception to P&C modernization: insurers often have many lines of business, often adjust rates/coverage, and are frequently dependent on reinsurers, brokers, payment processors, and regulators--so need to be able to change quickly and at the same time in a controlled manner.

Security and compliance are also at the center stage. Threats considering shared responsibility, virtualization, identity, data protection and multi-tenancy remain consistent in cloud security surveys [10]. Risk control lists like NIST SP 800-53 offer a systematic basis on the enforcement of security and privacy controls that are appropriate to a regulated environment, which is well aligned to the requirements of insurers of audit evidence and access governance, monitoring, and incident response [14]. Vendor lock-in literature also stipulates further that interoperability and portability should be considered as a design-time issue [12], [20].

In short, previous studies provide solid building blocks on the building blocks of cloud value propositions [1], [2], SaaS adoption dynamics, migration evidence, [4], [5], microservices transformation insights, [6], [7], delivery practices, [8], [9], and security frameworks, [10], [14]. The gap that is yet to be bridged in most practitioner modernization programs is the coherent, P&C-domain-based synthesis integrating architecture, migration strategy, and quality engineering into a single structure publishable in print-that paper fulfills that need.

3. Research Approach and Modernization Framework

The paper adheres to the design-oriented research methodology: the synthesis of peer-reviewed evidence and patterns that have been demonstrated to be successful by practitioners into a modernization framework, specific to the workload of P&C insurers. The studies on cloud migration show that companies are typically challenged by disjointed strategies, inconsistencies in terms of nomenclature (e.g., modernization vs. migration vs. reengineering), and insufficient migration tools to facilitate and prove migration execution [4]. We call such a collection of interventions here modernization, which is a set of interventions chosen per capability by the business criticality, regulatory constraints and architectural viability [5], [21] to include rehost, replatform, refactor, replace with SaaS or rebuild.

3.1 Capability-driven scoping

P&C can further be broken down into limited capabilities product configuration/rating, policy life cycle, billing/collections, claims intake and adjudication, document generation, customer identity and servicing, and analytics/fraud. In microservices decomposition studies, it is emphasized that selecting suitable service boundaries is a significant obstacle and cannot be done on an ad hoc basis but through conscious decision frames [7], [15]. We thus suggest that we begin with domain mapping (bounded contexts) and categorize each capability into:

- Differentiating vs. commodity: is the role an industry differentiator or an industry utility?
- Change frequency: how often is the rules, rates and workflow changed?
- Regulatory pressure: audit, retention and data governance.
- Integration density: upstream/ downstream dependencies number and criticality.

3.2 Target operating model selection

The evidence on the adoption of SaaS indicates that the perceived strategic relevance and uncertainty have a strong impact on adoption, and adoption varies in relation to the type of application [3]. Fully core SaaS replacement could be a good fit to certain business models (greenfield products, specialty lines) of the insurers, whereas hybrid architectures can be more useful in gradual modernization and risk reduction. The study of vendor lock-in supports creation of portability and exit plans when possible in key areas of core data and integration layers [12], [20].

3.3 Migration execution model

Based on literature on legacy-to-cloud migration, a proposed structure migration into Plan - Prepare - Migrate - Stabilize - Optimize, explicitly introducing quality phase gates to fill the gap in practice common with post-migration quality concerns revealed only at the end [5]. The stages consist of quantifiable deliverables:

- Plan: base architecture, dependency mapping and data classification [4], [5].
- Plan: reference architecture, security control mapping (e.g. NIST controls) toolchain readiness [14].
- Migrate incremental cutover patterns migrate by using strangler migration Pattern or modular replacement patterns, not big-bang migrations [21], [22].
- Stabilize: observability, SLOs, incident runbooks, performance verification [13], [16].
- Optimize: Optimize cost, optimize performance, optimize resiliency, optimize lock-in risk review [12].

3.4 Embedded quality engineering

The models of continuous delivery indicate that with benefits come difficulties, in particular, when quality checks are delayed or the non-functional requirements are not constantly verified [9]. The literature on DevOps also reveals that automation and cross-functional teamwork are required to maintain the speed of delivery without making the operations more risky [8]. Therefore, we incorporate quality engineering into the structure, with automatic testing, security testing, performance and resiliency testing and production telemetry feedback loops [8]-[11].

4. Target SaaS-Based Cloud Reference Architecture for P&C Platforms

The target architecture of P&C modernization based on SaaS has to strike a balance between configurability (product/rate changes), regulatory controls, integration extensibility, and operational resilience. The literature on cloud computing points out that cloud as utility alters the infrastructure consumption and scaling, however, insurers must require more than elasticity, they must have architectural separation of concerns that makes continuous change a reality with controlled risk [2]. On the same note, the business perspective work underlines that cloud value can be achieved where the organizations decide to match the technology changes to the governance changes and process evolution [1].

4.1 Architectural layers

Our suggested reference architecture is a layer architecture:

(A) Experience Layer: API consumer apps, agent and customer and partner omnichannel portals. RESTful homogeneous interfaces continue to be a prevalent style of integration with capabilities facing the outside world based on the scalability and interoperability features [19].

(B) Domain Services Layer: microservice/modular services, which are limited to specific bounded context (e.g. Claims FNOL, Coverage Verification, Billing Plans). The Microservices

research emphasizes that independent deployment and scaling are positive and need to be cautiously applied to cross-service consistency and visibility [7], [6]. Patterns catalogs offer useful advice on pattern of decomposition, communication and transaction patterns, which are especially applicable in claims/billing processes across multiple services [18].

(C) Policy administration SaaS, claims SaaS, billing SaaS, document composition SaaS, CRM SaaS. Evidence Studies on SaaS adoption indicate that the type of application has a strong impact on drivers in adoption; the SaaS adoption by the insurers must therefore be treated as capability-specific and not uniform to the enterprise-wide [3].

(D) Integration and Eventing Layer: API gateway, service mesh, event streaming, canonical data mapping. The idea of integration is identified as one of the common pain points of distributed migration works; event-based methods are able to decrease the synchronous links, yet it introduces schema management and idempotency demands [6], [16].

(E) Data Layer: polyglot persistence having explicit ownership (per bounded context), and an analytical platform. The data quality literature underlines the consumer-orientedness of data quality through the term of fitness-to-use, according to which the modernization of the insurers must specify the data quality dimensions on an individual use-case (claims fraud detection vs. regulatory reporting vs. billing accuracy) basis, as opposed to the adoption of an all-purpose measure [11].

(F) Security, Compliance, and Observability: identity and access management, audit logging, continuous monitoring and control mapping. The controls described by NIST SP 800-53 offer a broad list of controls applicable to the insurer context to be audit-ready in terms of privacy risk and to reduce privacy risks [14]. Other aspects of identity, virtualization, data confidentiality, and threat surfaces initiated by shared responsibility models are also highlighted by cloud security surveys [10].

4.2 Multi-tenancy and configurability

Multi-tenancy is often sparked to enable SaaS-based insurance platforms to scale in an economical manner, but tenant isolation and configuration safety are essential. The multi-tenant architectures need to provide rigid logical separation, encryption and authorization boundaries supported by product/rate configurations and workflow personalization [10], [30]. Regulatory segmentation (e.g., region/state rules) is often demanded by insurers to have separate environments or firm control by tenants.

4.3 Resilience and operational readiness

Distributed system architectures and microservices related architecture augment the significance of resilience designs (timeouts, backoff retries, circuit breakers) and end-to-end tracing [7], [17]. Evidence of continuous delivery suggests that the capability to not only release regularly with no incident rise is reliant on automation and efficient production feedback loops [9]. Hence, SLO-based observability, automated rollback, and resilience testing are first-class design requirements that are required in the target architecture [8], [17].

The reference architecture offers a consistent foundation to the migration strategy in the next section: it establishes the location of SaaS, custom differentiation, and what platform capabilities are needed to safely modernize.

5. Migration Strategy and Execution Plan

To migrate a P&C platform to a SaaS-based cloud target would need a P2C migration plan that minimizes business risk, continuity, and prevents the so-called hybrid paralysis, where legacy and new systems only develop and keep evolving without converging. The studies of systematic cloud migration reveal that the lack of mature tool support and standard execution frameworks is a common condition in many organizations, resulting in the uneven distribution of its results and the development of technical debt [4]. According to such recent reviews of the legacy-to-cloud, it is also noted that quality problems often arise after the migration due to the fact that the issue of quality are not properly addressed during the planning and implementation [5]. As a solution to that, we would offer a migration implementation strategy based on a staged delivery, gradual cutover, and quantifiable quality gates.

5.1 Selecting a migration path per capability

Our choice strategy is a capability-by-capability one:

- Use SaaS in case the functionality is commodity and in case the insurer is willing to take configuration-first customization (e.g., document generation, some customer servicing functions). The drivers of SaaS adoption vary depending on the type of application, hence, insurers are advised to focus on the SaaS where the uncertainty is not critical and the strategic value is evident [3].
- Strangler modernization of core transactional flows (policy/claims/billing): step by step send certain journeys to new platform and leave stable flows on legacy until confidence and coverage are achieved [21].
- Refactor to microservices in the absence of differentiation but in a situation where the legacy code can be broken down in a sustainable manner. Empirical migration research demonstrates that identification of services and organizational transformations are both major issues; the maturity of DevOps tends to be a condition [6], [8].
- Replatform: when the limiting factor is infrastructure/environment (e.g. batches of workloads or databases to managed services) with little functional modification.

5.2 Data migration and coexistence patterns

The hardest part of the insurers modernization is data due to long lifecycle of claims and policies, legal requirements of retention, and reliance on reporting. Research on data quality suggests taking quality as the aspect of fitness-to-use, where the migration is required to maintain properness in terms of consumer requirement like financial reconciliation, claims adjudication and statutory reporting [11]. Practically, this implies:

- Defining capability data contracts.
- Setting reconciliation controls (number of records, amount of premiums, balances of payment, balances of claims).
- Processing dual-write patterns or change-data-capture patterns when necessary and there is no alternative and monitoring is required.

The literature of microservices patterns offers guidance to data ownership and transaction patterns (e.g. saga orchestration), which are required to be adopted once monolithic ACID transactions are replaced by distributed workflows [18]. This particularly applies to claims payment and billing adjustments which demand uniformity across services.

5.3 Execution mechanics and validation gates

The migration is carried out in releases that are in line with continuous delivery concepts. Experience demonstrates that the continuous delivery has significant payoffs but needs a high level of automation and non-functional pipeline verification [9]. Thus, gates are contained in every migration increment, and include:

- Functionality right (automation-first regression).
- Performance (under peak loads, under baseline and target conditions, and during catastrophe events).
- Security (identity, encryption, vulnerability scanning based on the cloud security threats) [10].
- Technical availability (dashboards, alerts, runbooks, rollback plans) [8].

5.4 Avoiding lock-in and preserving optionality

The literature on vendor lock-in cautions that migration often increases dependency risks in case portability is not thought out in the beginning [12], [20]. In the case of insurers, mitigation on a practical level consists of abstraction of integration through APIs/events, core dataset exportable, and a formal exit process that is periodically tested. This will not make SaaS adoption an irreversible technical debt.

This plan prepares the following part: integrating quality engineering and DevSecOps in such a way that each increment can be tested, audited, and deployed.

6. DevSecOps and Quality Engineering for Modernized Insurance Platforms

The implementation of a SaaS-based modernization program would fail because an increase of delivery speed combined with a decrease in quality, security and compliance ensured. The literature of DevOps makes the purpose to complement development, delivery and operations to enhance throughput and reliability by automating and collaborating [8]. Ongoing delivery evidence also indicates that successful teams attain meaningful gains, yet they need to be able to deliver quality, push it further into the pipeline and not delay quality assurance to the end [9]. In the case of insurers, this principle will be transformed into DevSecOps-based quality engineering when the security, testing and compliance evidence is constantly generated.

6.1 Quality model aligned to insurance outcomes

The modernization of insurers involves other attributes of quality than functional correctness: they must be available during catastrophe spikes, they must have their bills correct, their claims cycle-time must be efficient, they must be traceable in case of audit, and their data must maintain integrity. The research on data quality focuses on the consumer-oriented quality dimensions (accuracy, timeliness, completeness, interpretability, etc.), which align with insurance applications (regulatory reporting, customer billing transparency) [11]. We suggest that a quality model be defined by each capability:

Claims: fraud- signal latency, correctness, auditability, availability.

- Billing: financial correctness, reconciliation, handling payments which are idempent.
- Policy rating correctness of rules, transparency of rules, and versioning.

6.2 Pipeline architecture and automated assurance

The layered quality gates, which should be used in the CI/CD pipeline, are:

- Code quality: static analysis, dependency scan, policy-as-code scan.
- Automated tests: unit, API/event contract test, U/SaaS sandbox tests (integration), as well as end-to-end journey tests (quote - bind - endorse - claim FNOL - payment).
- Non-functional testing: load testing, resilience testing and security testing.
- Release controls release controls canary release, feature flag, and automated rollback Release controls. Release controls are canary release, feature flag, and automated rollback.

The research in microservices shows that the distributed system raises the demand of contract testing and observability due to the inability to identify the failure mode through single system testing [7]. Literature on microservice patterns offers practical strategies on managing boundaries of service, reliability of communication, and workflow transactions (e.g., sagas) that should always be tested (e.g., via continuous validation) [18].

6.3 Security-by-design and control evidence

According to cloud security surveys, identity, data confidentiality, virtualization threat, and misconfiguration are common threats vectors [10]. To operationalize security in the modernization of insurance, we suggest that Home Depot maps pipeline controls and runtime controls to NIST SP 800-53 families (access control, audit/logging, incident response, configuration management, etc), and that the evidence is automatically generated using logs, scan reports, approvals, and immutable audit trails [14]. This makes compliance an incidental result of regular engineering effort and not a last-minute rushy report writing.

6.4 Measuring performance and outcomes

Recent studies of DevOps measurement associate delivery performance with organizational performance and offer empirically-based metrics of throughput and stability [23]. The use of these metrics can assist insurers in proving their value of modernization: a shorter lead time for

ratio adjustments, a higher deployment rate with no spikes, faster recovery, and fewer change failures. Such results are to be supplemented with domain KPIs (claim cycle time, policy issuance time, premium leakage reduction).

In brief, quality engineering implemented by DevSecOps is the engine of implementation that will enable the delivery of SaaS migration increments in a continuous manner with insurer quality reliability, traceability, and security.

7. Governance, Compliance, and Operating Model for SaaS Modernization

P&C modernization is no less of an operating model shift than a technology shift. The approach of cloud business perspective emphasizes that value creation requires the connection between technology adoption and organizational governance and decision rights [1]. Governance on the part of the insurers has to cut across vendor management (SaaS), architecture, security controls, data stewardship, and release management, frequently across multiple lines of business and geographies.

7.1 Target operating model

We suggest that there be three-layers of operating model:

* Platform governance: sets reference architectures, reusable security bases, common CI/CD tooling and observability requirements. The literature of DevOps focuses more on shared accountability between development and operations, and that means that platform teams must deliver self-service guardrails to product teams, rather than be gatekeepers themselves [8].

* Product/capability teams: are the owners of their small business outcomes and service reliability in their mini-contexts (claims FNOL, billing adjustments etc.), and are consistent with microservice ownership principles [7].

* Risk and compliance enablement: functions to integrate compliance mapping (tooling, audit evidence, data retention) in a process and eliminate manual bottlenecks.

The control mapping and audit readiness - This is to demonstrate how well the financial statements comply with the company's requirements.

7.2 Control mapping and audit readiness

his is to show how financial statements are prepared to meet the requirements of the company.

NIST SP 800-53 offers a standardized list of controls that can be used in the development of audit-ready programs, particularly those that focus on access control, monitoring, incident response, and configuration management [14]. The shared responsibility model in the context of SaaS modernization has to be explicitly defined: what are the controls that the SaaS vendor provides, what is configured by the insurer, what are the controls that have to be implemented in case of custom extensions. According to the surveys of cloud security, misconfiguration and identity vulnerabilities are reported to be frequent, so the governance should incorporate robust IAM patterns, privileged access control, and continuous configuration test [10].

7.3 Vendor lock-in and portability governance

The literature on the lock-in demonstrates that standardized interfaces, caution in contracting as well as architectural decoupling can reduce risks [12], [20]. Insurers should implement:

EEE Exit strategy playbooks (functional fallback, migration paths, data export).

* Critical capabilities (core data ownership, integration standards) Portability.

* SaaS risk scoring which incorporates operational resilience, compliance capabilities, and audit support requirements.

7.4 Data governance

Fitness-for-use should be the defining force of data quality, and data products and definitions have explicit policy, claims, and billing ownership [11]. This incorporates the master data alignment (customers, vehicles, properties), reference data versioning (coverage codes, rate tables) and processes of reconciliation during migration.

7.5 Production operations

Lastly, there should be operational maturity to architectural complexity. Microservices journeys indicate operational complexity to be a significant constraint that needs mature observability and incident management [7]. The repeated occurrence of delivery work drives the point that fast release demands consistent enough foundations to operate, such as monitoring, feedback loops, and environment consistency [9]. Governance must then stipulate SLOs, on-call preparedness, disaster recovery exercising, and chaos/resilience examination on critical services [17].

8. Discussion, Limitations, and Future Research Directions

The suggested strategy will be used to integrate existing evidence on clouds, SaaS, migration, and DevSecops into a P&C-specific modernization road map. The central point of migration literature is that the execution discipline and quality integration are the key factors influencing the success of modernization, not the adoption of the cloud only [4], [5]. The modernization challenge is exceptionally difficult in the case of P&C insurers due to the long lifespan of business processes (policies and claims may take years), the high governance of the data, and the high density of the ecosystems. SaaS target is able to lessen commodity maintenance overhead and expedite the provision of capabilities, yet it may introduce configuration constraints, intricacies of integration and portability constraints otherwise if not managed with caution [3], [12].

8.1 Practical trade-offs

Research on microservices indicates that decomposition enhances deployability and scaling yet has a negative effect on the complexity of distributed-systems and operational maturity requirements [7]. Microservices by default should therefore not be embraced by insurers. Rather, microservices are to be applied selectively to areas of the system that need autonomous change rate (e.g. digital claims intake, partner APIs, experimentation with dynamic rating), and

SaaS and managed services are to be adopted to minimize undifferentiated engineering labor. Pattern-based guidance assists this discriminative methodology by offering effective design strategies to service breaking and cross-service processes [18].

8.2 Quality and compliance as modernization accelerators

One of the key contributions of the paper is the considerations of DevSecOps quality engineering as one of the key enablers of modernization. Evidence of continuous delivery indicates that high performance requires the level of automation as well as early quality validation without which modernization may enhance the occurrence of failure and operational load [9]. In the case of regulated industries, compliance can be turned into a repeatable system by mapping the control into pipelines and runtime monitoring (e.g. NIST) [14]. The surveys of cloud security support the idea that identity, data protection, and misconfiguration are the areas of constant risk- i.e. control automation and constant monitoring is a must-not a should-have [10].

8.3 Limitations

The paper is a synthesised framework as opposed to a one-case empirical study. Although it is based on peer-reviewed evidence, the results will depend on the size of the insurer, the complexity of a product, its focus in a regulatory environment, and its vendor ecosystem. Also, the details of SaaS platforms vary greatly; one may offer deep extensibility and observability capabilities, whereas others are limiting. The lock-in risk, then, is context-specific, and it has to be considered in terms of vendors and architectures [12], [20].

8.4 Future research

This paper can be developed further in the future by:

* Carrying out comparative case studies of insurers modernizing using various SaaS-core strategies.

* Coming up with quantitative imperative of migration risk rating (data complexity, integration density, regulatory burden).

* Measurement of the multi-tenancy isolation behavior and insurance-specific compliance automation [30].

* 251) Exploratory research into the application of modern observability and resiliency engineering methods to claims and billing system reliability during catastrophe-scale load [17].

References

[1] Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing—The business perspective. Decision Support Systems, 51(1), 176–189. https://doi.org/10.1016/j.dss.2010.12.006

[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing.

Communications of the ACM, 53(4), 50–58. https://doi.org/10.1145/1721654.1721672 profsandhu.com

[3] Benlian, A., Hess, T., & Buxmann, P. (2009). Drivers of SaaS-adoption—An empirical study of different application types. Business & Information Systems Engineering, 1(5), 357–369. https://doi.org/10.1007/s12599-009-0068-xSpringer Link

[4] Jamshidi, P., Ahmad, A., & Pahl, C. (2013). Cloud migration research: A systematic review. IEEE Transactions on Cloud Computing, 1(2), 142–157. https://doi.org/10.1109/TCC.2013.10 DBLP

[5] Hasan, M. H., Osman, M. H., Admodisastro, N. I., & Muhammad, M. S. (2023). Legacy systems to cloud migration: A review from the architectural perspective. Journal of Systems and Software, 202, 111702. https://doi.org/10.1016/j.jss.2023.111702 ScienceDirect

[6] Taibi, D., Lenarduzzi, V., & Pahl, C. (2017). Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation. IEEE Cloud Computing, 4(5), 22–32. https://doi.org/10.1109/MCC.2017.4250931 notes.davidkopp.de

[7] Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., & Tilkov, S. (2018). Microservices: The journey so far and challenges ahead. IEEE Software. https://doi.org/10.1109/MS.2018.2141039 X-MOL

[8] Ebert, C., Gallardo, G., Hernantes, J., & Serrano, N. (2016). DevOps. IEEE Software, 33(3), 94–100. https://doi.org/10.1109/MS.2016.68 ResearchGate

[9] Chen, L. (2015). Continuous delivery: Huge benefits, but challenges too. IEEE Software, 32(2), 50–54. https://doi.org/10.1109/MS.2015.27 ResearchGate

[10] Singh, S., Jeong, Y.-S., & Park, J. H. (2016). A survey on cloud computing security: Issues, threats, and solutions. Journal of Network and Computer Applications, 75, 200–222. https://doi.org/10.1016/j.jnca.2016.09.002ScienceDirect

[11] Wang, R. Y., & Strong, D. M. (1996). Beyond accuracy: What data quality means to data consumers. Journal of Management Information Systems, 12(4), 5–33. https://doi.org/10.1080/07421222.1996.11518099 X-MOL

[12] Moghaddam, F. F., & Thakur, A. (2016). Critical analysis of vendor lock-in and its impact on cloud computing migration. Journal of Cloud Computing: Advances, Systems and Applications, 5(4). https://doi.org/10.1186/s13677-016-0054-z Springer Link

[13] Fielding, R. T., & Taylor, R. N. (2002). Principled design of the modern Web architecture. ACM Transactions on Internet Technology, 2(2), 115–150. https://doi.org/10.1145/514183.514185 BibSonomy

[14] Joint Task Force. (2020). Security and privacy controls for information systems and organizations (NIST SP 800-53 Rev. 5). National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-53r5 csrc.nist.rip

[15] Fritzsch, J., Bogner, J., Zimmermann, A., & Wagner, S. (2019). From monolith to microservices: A classification of refactoring approaches. In Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment (pp. 128–141). Springer. https://doi.org/10.1007/978-3-030-06019-0_10 Springer Link

[16] Newman, S. (2015). Building Microservices. O'Reilly Media. ISBN 9781491950357. VitalSource

[17] Basiri, A., Behnam, N., de Rooij, R., Hochstein, L., Kosewski, L., Reynolds, J., & Rosenthal, C. (2016). Chaos engineering. IEEE Software, 33(3), 35–41. https://doi.org/10.1109/MS.2016.60

[18] Richardson, C. (2018). Microservices Patterns: With Examples in Java. Manning Publications. ISBN 9781617294549. Simon & Schuster

[19] Hasan, M. H., Osman, M. H., & Admodisastro, N. I. (2023). From monolith to microservice: Measuring architecture maintainability. International Journal of Advanced Computer Science and Applications, 14(5). https://doi.org/10.14569/IJACSA.2023.0140591 ORCID

[20] Calinescu, R., Tudorache, T., & Kwiatkowska, M. (2014). A systematic review of cloud lock-in solutions. Proceedings / technical report (University of York repository). (DOI not consistently assigned in publisher record; use repository version for citation). PURE

[21] Fowler, M. (2004). Strangler Fig application. martinfowler.com (pattern description). (No DOI; industry pattern reference).

[22] Ross, J. W., Weill, P., & Robertson, D. C. (2006). Enterprise Architecture as Strategy. Harvard Business Review Press. ISBN 9781591398394.

[23] Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The Science of Lean Software and DevOps. IT Revolution. ISBN 9781942788331. Simon & Schuster

[24] Kitchenham, B., Brereton, P., Budgen, D., Turner, M., Bailey, J., & Linkman, S. (2009). Systematic literature reviews in software engineering—A systematic literature review. Information and Software Technology, 51(1), 7–15. https://doi.org/10.1016/j.infsof.2008.09.009

[25] Petersen, K., Feldt, R., Mujtaba, S., & Mattsson, M. (2008). Systematic mapping studies in software engineering. In Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering (EASE 2008)(pp. 68–77). BCS Learning & Development Ltd. https://doi.org/10.14236/ewic/EASE2008.8

[26] Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). Experimentation in software engineering. Springer. https://doi.org/10.1007/978-3-642-29044-2 (Print ISBN: 978-3-642-29043-5; eBook ISBN: 978-3-642-29044-2)

[27] Lenarduzzi, V., Lomio, F., Saarimäki, N., & Taibi, D. (2020). Does migrating a monolithic system to microservices decrease the technical debt? Journal of Systems and Software, 169, 110710. https://doi.org/10.1016/j.jss.2020.110710

[28] Gai, K., Qiu, M., & Zhao, H. (2016). Security-aware cloud computing: A survey. Future Generation Computer Systems, 67, 1–12. https://doi.org/10.1016/j.future.2016.02.020 Springer Link

[29] Mao, R., Zhang, H., Dai, Q., Huang, H., Rong, G., Shen, H., Chen, L., & Lu, K. (2020). Preliminary findings about DevSecOps from grey literature. In 2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS) (pp. 450–457). IEEE. https://doi.org/10.1109/QRS51102.2020.00064

[30] Nguyen, Q.-D., Pham, V.-N., Dang, T.-N., Phan, T.-N., & Phan, N.-M. (2025). Customer experience survey management software: A secure and scalable multi-tenant architecture. SN Computer Science, 6, 907. https://doi.org/10.1007/s42979-025-04433-z