AGILE–DEVOPS INTEGRATION IN REGULATED ENVIRONMENTS: CHALLENGES AND STRATEGIES FOR COMPLIANCE-DRIVEN INDUSTRIES

Lappeenranta–Lahti University of Technology LUT

Master's Thesis, Digital Systems and Services Development

2025

Sumaiya Tabassum Antara

Examiner(s): Professor Kari Smolander

Muhammad Hamza (Junior Researcher)

ABSTRACT

Lappeenranta-Lahti University of Technology LUT

LUT School of Engineering Sciences

Software Engineering

Sumaiya Tabassum Antara

Agile-DevOps Integration in Regulated Environments: Challenges and Strategies for Compliance-Driven Industries

Master’s Thesis

2025

77 pages, 07 figures, 08 tables and 03 appendices

Examiner(s): Professor Kari Smolander (LUT) and Junior Researcher Muhammad Hamza (LUT)

Keywords: Agile-DevOps, Regulatory Compliance, CI/CD Pipelines. Compliance-as-Code, Governance-by-Design, Compliance Automation, Regulated Industries, Constraint-Practice-Evidence framework

Strict regulatory controls in banking, healthcare and energy sectors make it hard for their software development and operations teams to fully implement Agile and DevOps practices. This study explores how organizations combine mandatory compliance with iterative software delivery. With interviewing a number of senior practitioners from banking and energy sectors and systematically analysing thirteen published industry cases, this research finds out the common constraints, adaptive practices and evidence mechanisms. All data were analysed using a unified Constraint-Practice-Evidence (C-P-E) coding framework to identify the recurring patterns across different sectors. The findings present four dominant regulatory constraint categories which are mitigated by automating compliance directly into CI/CD pipelines, integrating compliance tasks into sprint planning and enforcing policies through code and service accounts. The resulting Constraint-Practice-Evidence framework shows that agility and regulatory control can coexist when compliance is integrated as automated rather than being a separate manual process. Overall, this research offers practical guidelines for DevOps teams along with suggesting a clear and short Constraint-Practice-Evidence model.

ABBREVIATIONS

CAB	Change Advisory Board
CI/CD	Continuous Integration/Continuous Deployment
C-P-E	Constraint-Practice-Evidence
DAST	Dynamic Application Security Testing
DevOps	Development and Operations
DevSecOps	Development, Security and Operations
DevCertOps	Development, Certification and Operations
DORA	Digital Operational Resilience Act
DPIA	Data Protection Impact Assessment
GDPR	General Data Protection Regulation
IAM	Identity and Access Management
IEC 62304	Medical Device Software–Software Lifecycle Processes
IEC 62443	Industrial Automation and Control System Security
ISO 9001	Quality Management Systems-Requirements
ISO 13485	Medical Device- Quality Management Systems
ISO 27001	Information Security Management Systems
MiCA	Markets in Crypto-Assets Regulation (EU)
MLOps	Machine Learning Operations
OPA	Open Policy Agent
PII	Personal Identifiable Information
QMS	Quality Management System
RBAC	Role-Based Access Control
RegOps	Regulatory Operations

SAFe	Scaled Agile Framework
SAST	Static Application Security Testing
SBOM	Software Bill of Materials
SoD	Segregation of Duties

ACKNOWLEDGEMENTS

I want to express my heartfelt gratitude to everyone who supported me throughout this Master's thesis journey. First and foremost, all praise and gratitude belong to the Almighty Allah for giving me the strength, patience and ability to complete this Master's thesis.

My deepest thanks go to my supervisor, Professor Kari Smolander for his continuous encouragement, clear guidance and precious feedback. Despite his extremely busy schedule, he made time for so many meetings with me over the past five months to guide me in the right direction and help me to shape this work into something I can be truly proud of.

A huge thanks to the five industry experts who generously agreed to give their valuable time and share real-world experiences during the interviews. Their honest insights from daily work in production gave this research a new dimension and real strength.

A big thanks to my friends and fellow students at LUT University for the countless discussions, moral support and coffee breaks that kept me going during those hectic days when I was endlessly working hard to analyse the data and find out a practical and meaningful research outcome.

To my beloved husband, thank you so much for supporting me every single day and every single moment. You always kept motivating and pushing me to give my very best. You kept me positive whenever I got exhausted and never let me lose hope. This thesis carries your love on every page.

I am forever grateful to my parents. Because of your sacrifices and belief in me, I was able to come to Finland and pursue my dream Master's degree. Your continuous love and prayers gave me strength from thousands of miles away.

Finally, thank you LUT University for giving this wonderful opportunity and for providing such an amazing environment that helped me grow.

Table of Contents

List of Figures

List of Tables

DECLARATIONS

Turnitin

The originality of this thesis has been reviewed with the Turnitin similarity checking service.

AI Usage

The author of this thesis, Sumaiya Tabassum Antara, used the following AI tools during the preparation of the thesis.

1. Grammarly AI tool
   a. Purpose of Use: Language maintenance.
   b. Explanation of the use of this tool: Throughout the document, grammar and spelling checking have been done using the Grammarly AI tool.
2. ChatGPT and Grok AI Tool
   a. Purpose of Use: Understanding the concepts and ideating the structure of research work.
   b. Explanation of the use of the tool: Searching for information within constraints, understanding the concepts and testing own knowledge. Ideating the structure of the research work.

NOTE! Record all the AI tools you have used in the same way.

Responsibility

The author, Sumaiya Tabassum Antara, takes full responsibility for the content of this thesis and has reviewed and edited the content generated by the possible use of AI tools.

1. Introduction

Agile was introduced in 2001 with the Agile Manifesto which promotes small and frequent deliveries and steady customer feedback (Beck et al., 2001). The DevOps concept emerged in the late 2000s as a culture and toolkit that brings development and operations together (Bass et al., 2015). It uses automation with CI/CD pipelines, testing and infrastructure as code in order to move from code to production safely and repeatably. In today's fast changing technology world, organizations want to deliver software quickly and reliably using Agile and DevOps methods. Together, the aim of Agile and DevOps practices in the software development lifecycle is to ensure fast and reliable delivery by combining iterative planning with automated and stable release workflows.

But in highly regulated sectors like banking, healthcare, energy and FinTech, the strict compliance rules make it hard for the teams to use these methods (Fitzgerald et al., 2017; Moyón et al., 2020). Regulation is an essential factor in these sectors to protect money, patients and critical infrastructure which ensures safety, fairness, privacy and system stability. The rules like GDPR for data privacy, DORA for digital resilience in finance and ISO standards for safety in healthcare sectors require careful checks, approvals and records. However, these regulations often clash with the iterative and continuous workflows of Agile and DevOps methods which makes the development and operation teams to fully implement them (Fitzgerald et al., 2017; Laukkarinen et al., 2017). For example, the developers do not get direct access to production systems due to segregation of duties (SoD), every change needs Change Advisory Board (CAB) approval and teams get huge pressure on creating excessive documentation before audit. These hurdles and extra works usually extend the development cycle and slow down the releases. And this tension is the core problem that this thesis addresses.

Prior works have identified many of these challenges and also shown that Agile and DevOps can work in regulated settings. Fitzgerald and Stol (2017) explained the fundamental tension between the speed of continuous delivery and the need for control in regulated environments. Security standards like IEC 62443-4-1 slow down the pipelines and require a lot of extra manual checks which have been addressed by Moyón et al. (2020, 2021). Research also shows that medical device software must meet the regulation of having all activities complete before release while keeping short feedback loops (Lie et al., 2016; McHugh et al., 2015). All these challenges lead to delays and reworks because the regulatory requirements are still being treated as a separate phase in most organizations and this issue has been pointed out by Poth et al. (2019) and Rahman et al. (2019).

Despite having these valuable insights from previous research, there are some clear gaps which leave a field for further empirical research works. Most of the studies focus on single sector like finance or healthcare and a single standard like IEC 62443 or ISO 13485 pointing to some common blockers like release completeness rules, segregation of duties, change windows and traceability gaps. The lack of cross-sector empirical evidence is seen. Very few works systematically connect the specific regulatory constraints to the adaptive practices to mitigate them and the exact evidence that the auditors accept. There is also limited practitioner feedback and perspectives combined with documented industry cases. This thesis aims to address these gaps by collecting and comparing real world data across multiple regulated sectors.

The aim of this study is to explore and understand how the regulated organisations successfully combine Agile-DevOps speed with strict regulatory compliance. After exploring the most practiced integration methods and adaptations, the goal of this study is to create a practical framework that shows how this main aim can be achieved. Three research questions guide the whole research process:

RQ1: What specific constraints limit the implementation of Agile and DevOps practices in compliance sensitive industries?

RQ2: How do teams adapt their processes, tools and culture to meet both agility goals and regulatory demands?

RQ3: Which practical pathways successfully link regulatory constraints to the right practices and acceptable evidence?

A qualitative design approach has been used in this research where semi-structured interviews are the main data source. Five interviews were conducted across banking and energy sector with software delivery team leads, DevOps leads and an auditor.

Additionally, a focused review of case studies serves as the secondary data source to strengthen the patterns and test transferability. All sources are coded with one simple scheme which is constraint, practice and evidence. The same coding is used for interviews and case studies so that they can be compared directly. Patterns that appear in more than one source are treated as stronger patterns.

In the next phase of this research, the analysis of the structured data produces three types of outputs that answer the three research questions. First one is a categorized list of most common constraints that slow down or reshape the delivery. Second one is a clear list of practices that teams actually use against those constraints along with any signs of effect. And third one is a framework of constraint, practice and evidence pathways. The proposed C-P-E framework offers practical guidelines to the teams to choose a path and know which proof to keep. It also helps the auditors to see how CI/CD can carry control and it does not weaken the control.

The main intention of this research is to show that compliance and agility can coexist when compliance is built into the technical pipeline rather than adding it as a separate manual layer. This process helps the teams move faster with confidence and assists the auditors to assess the system regulations faster with clarity. The scope of this study is not any kind of legal analysis of every standard. Rather, the scope is delivery workflows and pipeline evidence. The limitations from the small interview set are handled by transparent reporting of methods and decisions, careful coding and triangulation with well-described cases. The context of constraint, practice and evidence pattern has been adapted from the governance-mapping logic in Harmonizing COBIT & Scrum, 2020, which demonstrates how specific COBIT control constraints can be translated into Scrum-level practices and backed by auditable artifacts produced in the pipeline.

The rest of the thesis is organized as follows. Chapter 2 reviews the related research to establish a solid background for this study. Chapter 3 presents the research method. Chapter 4 reports the analysed results from interviews and case studies. Chapter 5 discusses the findings of the research and answers the research questions. Chapter 6 concludes the whole thesis work with contributions and suggestions for future research.

2. Literature Review

This chapter is about what prior studies say regarding Agile and DevOps in regulated environments. The background research focuses on three things which include the typical struggling points, helpful practices for teams and few metrics and artifacts that prove controls are working. This sets the ground for my study's own constraint-practice-evidence approach across banking, healthcare or medical devices and energy sectors.

2.1 Overview of Agile and DevOps in Regulated Settings

Agile is a way of developing software with small and frequent increments, close customer feedback and a focus on working software over any kind of heavy paperwork. DevOps extends this by having development and operations work together by using automation with CI/CD, tests and infrastructure as code so that changes can move from code to production reliably and repeatably. In regulated industries like banks, healthcare or medical devices and energy or critical infrastructure, the software teams must utilize these Agile-DevOps benefits along with meeting rules on safety, security, privacy, approvals and audit trails. The positive takeaway from the prior works is that most standards do not depend on the model used. Most of the Agile or DevOps practices are acceptable if we maintain traceability, conduct risk-based testing and ensure all decisions are documented throughout the process. (Laukkarinen et. al., 2017; Hrgarek et. al., 2012). It practically means building compliance into the flow. Instead of waiting to do the documentation at the end of the project, approvals need to be captured, results need to be tested or scanned and links need to be established from requirements to code, code to tests and tests to releases inside the pipeline. Medical-device studies call this out explicitly under ISO 13485/IEC 62304 (Laukkarinen et. al., 2017; Lie et. al., 2020). Besides, GDPR case work shows how privacy-by-default can be wired into CI/CD (Leite et. al., 2022). Moreover, bank case studies show DevOps working under segregation-of-duties and change windows (ICSME 2023) and COBIT vs Scrum work shows how to map governance controls to Scrum artifacts and events so that audits can rely on everyday Agile evidence (Harmonizing COBIT & Scrum, 2020). A forward-looking view in safety-critical domains even treats the pipeline itself as part of compliance which is defined as DevCertOps. It also includes tool qualification and auditable gates (Zeller et. al., 2023). All these research points to one practical goal directing to maintain a fast Agile and DevOps pace by ensuring the automatic creation of compliance evidence that the regulators need (Wagner & Ford, 2020).

Banking provides a strong reference point for these issues. A recent case study from a large financial organization shows how a team moved from manual handovers and no version control to a standard CI/CD template, coverage gates and automated deployments. The results were faster and provided more reliable releases, but the organization still had to respect strict separation of duties and production access rules, which required careful role design in the pipeline and stronger communication with operations (ICSME 2023). A separate study on COBIT vs Scrum in a major bank explains the governance side of that story. Conflicts of the steering committee vs product owner, independent testing, document approvals and role definitions can be resolved by placing approvals at the right level. And this is often done by placing it during the release instead of every sprint. Also, keeping independent testing before production and mapping COBIT objectives to Scrum artifacts and events, can also provide solutions for the conflicts (Harmonizing COBIT & Scrum, 2020). These results show that Agile teams can keep their pace if governance is integrated into the workflow rather than imposing it externally.

Privacy regulations have also pushed engineering to change how and when evidence is produced. A multi-case study on GDPR describes the concrete shifts across the whole lifecycle by mentioning privacy by design or default at requirements time, data mapping and DPIAs during analysis, encryption and pseudonymization choices during design, training and least-privilege during development, synthetic or masked data during testing and 72-hour breach-response and rights handling after deployment (Leite et. Al, 2022). The same study explicitly notes that several of these checks can be wired into a DevOps pipeline such as using gates to block releases without DPIA evidence for personal data features or jobs that generate masked test data. This is a good example of the general pattern in regulated settings which allows to move the checks into the flow and automate whatever is possible when the bar cannot be lowered (Leite et. Al, 2022).

Another practical concern in regulated work is documentation. Agile teams are often afraid by thinking that standards will force heavy and up-front documents. Multiple studies argue that this is a false dilemma if teams capture the right records from normal Agile work. A case on ISO 9001 shows how a Scrum-based process can satisfy Clause 7 -Product Realization by mapping user stories, reviews, tests, approvals, release notes and version control logs to the exact records that the standard expects. The authors even built an internal audit checklist and a quality manual around this mapping (Baloun, 2013). A follow-up study in a healthcare product company looked at 23 common documents used with Scrum and found that only a subset is actually useful across roles and many sections in large templates go unread. Simply, their recommendation was to keep fewer and clearer documents, make purpose and approvals explicit, put verification or validation evidence where it belongs and update documents during the sprint instead of after the fact (An Evaluation of Documentation, 2022). In short, Agile is equal to no documentation means making right-sized documentation at the right time.

There is also a people and governance angle which is especially important for small teams. A survey across small software companies reports that oversight, accountability and control practices are present everywhere, but they are quite uneven with the smallest companies struggling the most. The proposed fix for this issue is lightweight governance which has clear roles, routine lessons learned and basic contingency planning for small teams (Organizational Governance in SSCs, 2022). A standard comparison for very small entities (ISO/IEC 29110) found that UPEDU (a lightweight process derived from RUP) aligns very well with the standard's project-management expectations, Scrum aligns moderately and XP aligns poorly for project-management needs unless a small assessment or control loop and a few clear records are added (Galván-Cruz et. Al, 2017). These results support the broader idea that small organizations can stay Agile and still meet standards by adding a few targeted controls instead of establishing a full heavyweight process.

People often think that being secure or following regulations means that the work needs to be slower. But studies show that the rules can be followed without blowing up the entire process. For example, one study of an Agile company with 300 employees found that becoming compliant only needed some small changes (Bakalova et al., 2014). The main things they added were a rule of no change without testing and creating just enough documentation (Bakalova et al., 2014). The company saw great results with happier clients, better quality, fewer bugs and less rework. While the documentation sometimes meant that the project took a bit longer at first, the overall business value was huge (Bakalova et al., 2014). Because of this need to blend compliance into the work, other research says that relying only on paperwork should be stopped. Instead, small and measurable clues that show the controls are actually working, need to be focused on (Wagner & Ford, 2020). Important things to track include how many bugs are there, how much code is covered by automated tests and how often security tests pass. These numbers are helpful for auditors and for daily teams and they prove that rules are being followed as the work gets done (Wagner & Ford, 2020).

A coherent picture can be visualized by putting all these points together.

Firstly, standards are usually model agnostic. IEC 62304, ISO 13485/14971, ISO 900 and GDPR do not ban Agile and DevOps. They ask for repeatability, risk-based verification and validation, traceability, documented decisions and the ability to explain what changed, why and with what effect (Hrgarek et. Al, 2012; Baloun, 2013; Leite et. Al, 2022).

Secondly, pipelines are the natural place to carry this load. If approvals can be captured, results can be tested or scanned and links can be traced during builds and releases, then last-minute paperwork can be avoided. Also, a consistent story can be showed to the auditors (Zeller et. Al, 2023; Laukkarinen et. Al, 2017; ICSME 2023).

Thirdly, governance needs to be mapped to Agile artifacts and events. Scrum roles and ceremonies do not conflict with governance if they can be connected to control objectives like independent testing before production, release-level approvals, clear role separation in PRs and deploy steps and a Definition of Done that includes the right evidence (Harmonizing COBIT & Scrum, 2020; Baloun, 2013).

Fourthly, privacy and security are continuous and they are not one-time gates. Privacy by default belongs in requirements and analysis, synthetic data and deletion belong in testing and incident response and rights handling belong in operations. Security checks should run on every change and on a schedule with version-pinned policies to make sure that the results are always reliable (Leite et. Al, 2022; Wagner & Ford, 2020).

Finally, right-sizing is the key. Small teams need a few added controls for oversight, traceability and closure. Large organizations need standard pipelines, enabler coaching and clear role definitions to respect segregation of duties without losing speed (Organizational Governance in SSCs, 2022; ICSME 2023).

2.2 Typical Challenges

Many regulated teams face the conflict of release completeness with frequent delivery during release time where policy or standards say that all activities must be completed before a release is approved. This protects users and the organization, but it clashes with the Agile habit of releasing in small steps. When every increment must carry full documents, formal sign-offs and risk evidence, teams either slow down or build lots of paperwork after the spirit. This ultimately creates delay and audit risk. Medical-device case work shows this explicitly denoting rules like IEC 62304 expect completeness and repeatability and teams who do not plan for this end up with mini-waterfalls and long approval queues at the end of an iteration (Laukkarinen et al., 2017). Industry guidance makes the same point saying agile is acceptable in regulated contexts, but you must show the required evidence for each release, not only at the project end (Hrgarek, 2012).

Challenge also appears on unit acceptance before integration conflicting with being always-on CI. Continuous integration encourages teams code merge code early and often. Some safety-critical rules expect unit-level acceptance before integration. If it is handled as a manual phase like this, it blocks the CI rhythm. Teams then treat unit verification as a separate mini-stage and push integration later. But it compresses testing and brings lots of rework. The result is team slowdowns and discovering bugs too late. But the continuous integration (CI) system should have flagged those bugs much earlier (Laukkarinen et al., 2017). The core problem is not the regulation. It is how the regulation is being implemented. If teams cannot automatically show that new code passes acceptance tests before merging, they are forced to slow down.

Auditors and assessors often look for a clean line from requirement to code change, to tests, to build and finally to the deployed release. Agile teams spread this work across multiple systems including work trackers, source control, CI servers and deployment tools. Without automation, keeping those links current is manual and error prone. Many teams fall back to spreadsheets or ad-hoc screenshots, which are fragile and hand to maintain under audit pressure. Reviews of DevOps in regulated environments and ISO-aligned case work both list traceability gaps as a top pain point. They also note that evidence by construction is the practical alternative to late and document-heavy work (Lie et al., 2020; Baloun, 2013). Another big challenge is maintaining segregation of duties in small squads. Regulations and control frameworks except for independent approvals and testing. Scrum favours one cross-functional team where members wear multiple hats. In small squads, the same engineer may write, review, test and deploy a change unless the setup is adjusted. Banks add an extra constraint clearly stating no developer access to production. This lengthens hotfix paths if roles and pipeline steps are not designed carefully. Studies of COBIT with Scrum and a large-bank DevOps adoption both describe this tension and show that teams need explicit separation in pull-request reviews, pre-production testing and release approvals to satisfy independence without losing the flow (Harmonizing COBIT & Scrum, 2020; ICSME 2023).

Then comes the challenge in trust on tools and pipelines. In safety-critical and highly compliant work, auditors often ask whether the CI server, test harness and security scanners are fit for purpose. They inspect on how their versions are managed. If teams cannot answer these questions, automation gets questioned and manual re-checks creep back in. The DevCertOps view makes this explicit by treating the pipeline as part of the compliance system, qualifying critical steps and keeping versioned and auditable configurations (Zeller, 2023; Lie et al., 2020). Moreover, agile teams often fear that compliance means heavy documentation. Producing large templates with many unused sections and writing documents after the sprint just for compliance causes the main problem. Those habits generate inconsistent records that please no one. Case work shows two practical observations. The first one is, only a subset of documents is truly useful across roles and the second one is, documents work when they are updated during the sprint and tied to verification and validation evidence which are not written later under deadline (Baloun, 2013; An Evaluation of Documentation, 2022).

Skipping security testing and deep reviews is not an option in regulated contexts. The challenge is to split checks sensibly with faster run, automated checks on every change and scheduling deeper testing. Guidance aimed at regulated programs recommends pairing this with clear pass or fail thresholds and visible results per pipeline run so that teams do not debate every finding by email after the fact (Wagner & Ford, 2020; Leite et al., 2022). Then the legacy platforms and uneven standardization in regulated organizations bring a big challenge. Large banks and utilities often start from manual packaging, shared servers and fragile deployment paths. Even after adopting version control and CI/CD, teams face access hurdles, inconsistent logging and one-off pipeline scripts per application. These realities slow standardization, make onboarding harder and weaken the benefits of automation. A large-bank case study found that creating a standard, customizable pipeline template and an internal enabler team improved consistency, but the progress still depended on clearing legacy bottlenecks and planning time for adoption work (ICSME 2023).

Governance frameworks define steering committees, formal approvals and independent QA. Scrum includes Product Owner, Scrum Master and a development team. If organizations do not explicitly map these worlds, teams get mixed signals regarding who decides priority, who approves the releases and who signs off testing. Then adults end up asking for artifacts that Scrum does not produce by default. Bank experience with COBIT and Scrum shows that aligning decision rights and moving some approvals to the release level reduces friction without diluting control (Harmonizing COBIT & Scrum, 2020). Besides, collecting too many metrics or the wrong ones, creates noise and wastes attention. Leaders and auditors look for a small and stable set of signals that show controls are effective which includes defect trends, basic automated coverage, security test pass rates, detection rates from scanners and time to fix critical issues. The literature stresses on choosing measures the pipeline can produce reliably and discussing them regularly so that they can guide decisions rather than just remaining decorated dashboards (Wagner & Ford, 2020).

Finally, no technical setup can survive without people adopting it. Moving from ticket-based handovers to standard pipelines, code reviews, coverage gate and shared definitions of done requires new habits and skills. Resistance appears when benefits are not visible, when DevOps work is not planned into sprint capacity or when coaching and management support are missing. Reviews of regulated DevOps adoptions repeatedly highlight leadership sponsorship, an internal enablement group and time for teams to learn as the non-negotiables of change (Lie et al., 2020; ICSME 2023).

2.3 Common Enablers and Practices

A clear pattern in the literature is to put compliance inside the flow of work. Teams do this by making the pipeline produce evidence automatically. The pipeline automatically captures test results, security-scan reports and release notes, logs who approved and when at deployment and links each change back to its original story. When evidence is created as part of the build and release, audits no longer depend on manual documents written later and the team are able to maintain its delivery rhythm (Lie et al., 2020; Wagner & Ford, 2020).

Another enabler is lightweight release gates. Deployment should not be done unless a few must-have checks pass and the approver is recorded. Typical checks are like tests are green, no high-severity security findings and required documents or risk items are closed. The gate's own log becomes the audit trail. In safety-critical work, this aligns with the need for all activities to be complete before release. But it doesn't turn sprints into mini waterfalls (Laukkarinen et al., 2017; Hrgarek, 2012). In bank settings, gates also help to fit change windows and separation-of-duties rules into CI/CD, because approvals and roles are visible in the pipeline (ICSME, 2023; Harmonizing COBIT & Scrum, 2020). Teams also invest in repeatable environments. Infrastructure-as-code and containers make it possible to promote the same signed artifact through test, UAT and production. It satisfies repeatable release expectations and reduces last-mile risk (Laukkarinen et al., 2017). Alongside this, teams maintain a single trace link in all steps from requirement to release. They do this by carrying a unique ID through the branch name, pull request, build metadata and release notes. Small scripts or bots add the links, so the chain is automatic rather than hand built (Lie et al., 2020; Baloun, 2013).

Where governance frameworks meet Agile, the successful pattern is mapping control objectives to Scrum artifacts and events. Decision rights from steering forums flow into the Product Owner's backlog. Independent testing happens before production either by a different person in the team or by a central QA. Approvals move to the release level instead of every sprint. This keeps independence and documentation where auditors expect them, while the team keeps short iterations (Harmonizing COBIT & Scrum, 2020). A large bank's DevOps adoption shows the same. It has a standard and customizable pipeline template and an internal DevOps enabler team. That team helps the other teams in adopting the pattern one application at a time with clear coverage gates and one-click deploys (ICSME, 2023). Besides, documentation becomes lighter and more useful when it is updated during the sprint and tied to verification or validation. Studies show that only a subset of documents is needed across roles. Making purpose and approvals explicit and placing V&V evidence where the work happens, cuts rework and improves audit readiness. A simple way to operationalize this is to map Agile artifacts to ISO requirements, then use that map for internal audit checklists and a short Quality Manual. It is not adding paperwork, it is in fact pointing auditors to the outputs that teams already produce. (An Evaluation of Documentation, 2022; Baloun, 2013). So, two organizational practices appear repeatedly.

First, leadership sponsorship and coaching. Regulated DevOps adoptions that succeed have visible support, time set aside for rollout and access to coaches or an enablement group (Lie et al., 2020; ICSME, 2023).

Second, right-sizing governance for small teams. This adds a brief plan-vs-actual check each iteration, keeps simple closure notes and clarifies who approves what. This closes most compliance gaps without a heavy process (Organizational Governance in SSCs, 2022; Galván-Cruz et al., 2017).

Finally, several papers stress a minimal evidence pack per build. This pack lives with the artifact and typically includes test summaries, security scan results with tool or version, the approver’s name and timestamp, a short release note, an artifact hash or SBOM and the trace link back to requirements.

This small bundle is enough to defend a release in audits and to keep the flow fast across sectors (Wagner & Ford, 2020; Baloun, 2013; Leite et al., 2022; Zeller, 2023).

2.4 Indicative Metrics and Evidence

All the studied articles strongly suggest tracking a small and stable set of signals that the pipeline can produce every time. The goal is not to achieve more numbers. It is to have the proof that controls are working and make sure that each release is repeatable and safe (Wagner & Ford, 2020). Teams rely on few trends pulled from CI for ensuring quality. Defect trend (down over time) and basic automated test coverage (steady or rising) are one of them. These are simple to collect and help explain why a release is ready (Wagner & Ford, 2020). In the DevOps adopting banks, adding a coverage gate in the pipeline raised discipline and made releases more predictable (ICSME 2023).

For security, the literature suggests having checks that can be shown per build. Security test pass rate (SAST/DAST/dependency scans), detection rate for code scanners (are they catching seeded issues), and time to fix critical vulnerabilities need to be shown for ensuring security (Wagner & Ford, 2020). Also, keeping scanner policy versions pinned makes results repeatable and auditable across releases (Wagner & Ford, 2020).

For flow or control, the useful signals are change-failure rate, rollback rate and lead time to release. These need to be paired with evidence where each release is repeatable having the artifact hash/SBOM and the approver’s name and timestamp recorded by the pipeline (Wagner & Ford, 2020; ICSME 2023). ISO-oriented case work shows that mapping these pipeline outputs to required records reduces audit preparation time, because the document is already produced by normal work (Baloun, 2013).

Another crucial metric is privacy and GDPR studies recommend wiring operational metrics into backlogs and CI. It can be like a percentage of stories tagged as personal data with DPIA status, percentage of tests using synthetic or masked data, test-data cleanup success and breach-notification readiness (Leite et al., 2022). Authors emphasize certain things for the safety for regulated evidence. Those are traceability completeness, tool or pipeline trust and assurance bundles linked to each change in safety-critical work (Zeller et al., 2023; Laukkarinen et al., 2017; Lie et al., 2020).

Across all domains, many articles recommend keeping a minimal evidence pack per build with a short test summary, security scan results with tool and policy versions, approver and timestamp, a brief release note, artifact hash/SBOM and a trace link back to the requirement. Storing this pack with the build gives auditors a clear and repeatable story without slowing the Agile-DevOps flow (Wagner & Ford, 2020; Baloun, 2013; Leite et al., 2022; Zeller et al., 2023).

2.5 Chapter Summary

All this previous research clearly shows that regulatory requirements create many issues for Agile and DevOps adoption in compliance-driven industries. The reviewed literature highlighted the common constraints and suggested several adaptations. However, most of the studies focused on only one specific sector which included either banking or healthcare or industrial systems. Also, only one specific standard was mentioned in each of the reviewed literature. Very few previous studies compared practices across different regulated environments. Moreover, the existing research rarely combines real practitioner experiences with documented industry cases and any practical frameworks were not seen which directly link the specific constraints, exact adaptive mechanisms and evidence accepted by auditors. This thesis fills these gaps by performing qualitative research with interviews and case studies. The following chapters present the development of identifying the cross-sector patterns and show how compliance can be integrated into automated pipelines instead of remaining a slow and manual layer.

3. Research Method

This study has used a qualitative and interpretive approach where the main data source is semi-structured interviews with practitioners and auditors in banking sector, with comparison insights from healthcare and energy sectors. A secondary source is a set of case-based papers that were skimmed and summarized to capture the concrete practices and evidence. Interviews and cases were read together to understand constraints, the practices used to handle them and the evidence that supports audits. Findings are built from patterns that appear across both sources and are reported with clear and practical language.

3.1 Research Design Overview

For understanding how Agile and DevOps operate under regulation, I have used an interpretive and qualitative design. The unit of analysis is the software delivery team and its CI/CD pipeline within a regulated organization. The primary setting is banking and cases from healthcare and energy sectors are used for comparison. Semi-structured interviews serve as the main data source. I have conducted the interviews across key roles of audit and assurance, development team lead and DevOps and SRE team using open questions that could bring several examples. The sessions were recorded with their consent which were later transcribed and anonymized. As secondary data, I have selected case studies and peer-reviewed practice papers because they describe real activities, controls and artifacts rather than only concepts. All material in interviews and cases is coded in an Excel file with a constraint-practice-evidence scheme. The idea of this scheme was developed inductively during the analysis of raw interview and case study data. The consistent sequence of constraint, practice and evidence was observed repeatedly across different sectors and companies. So, these three categories were chosen as the organizing framework for the entire data analysis. The thematic coding started with a small list of known themes and was then opened to new ideas that arose. Patterns were cross-checked across different sources and sectors to see what repeated and what differed. Attention was given to trustworthiness through clear coding notes, versioned codebooks and short factual checks with informants when needed. The small interview sample was addressed by combining it with detailed case material and by focusing on pathways that appeared in more than one source. Ethics were observed through informed consent, secure storage and removal of identifying details. Figure 1 below shows the complete view of the research design from research questions to the final integrated results.

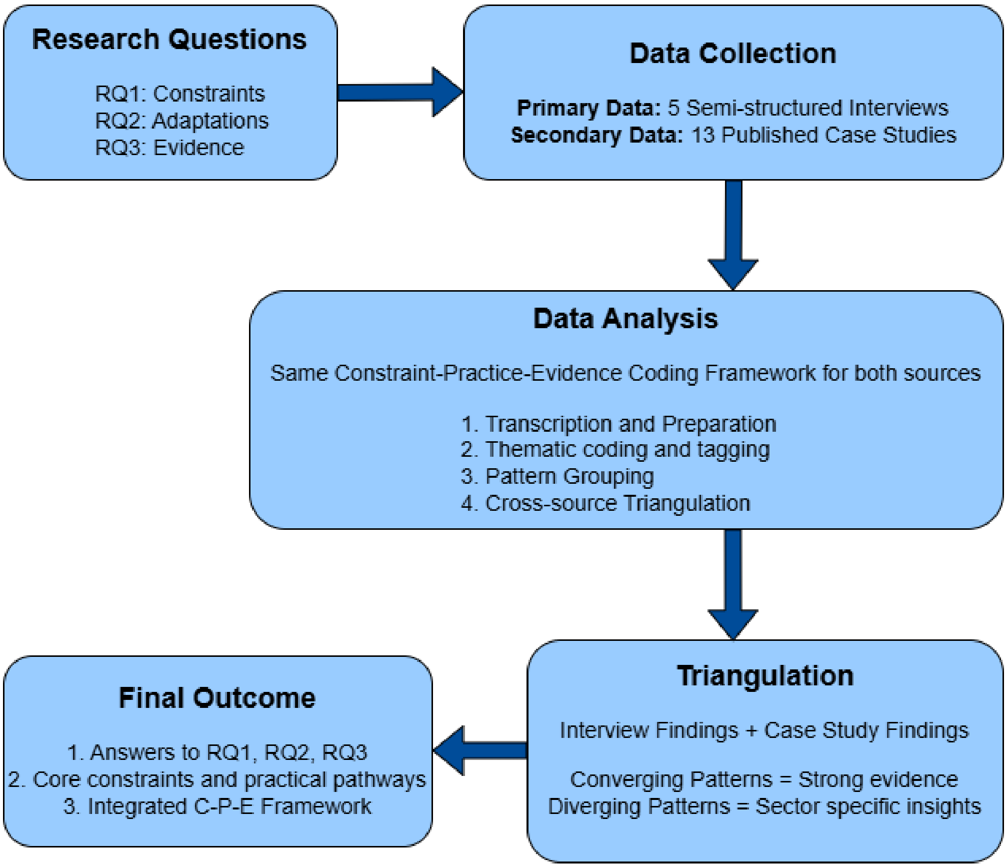


Figure 1: Overall Research Design and Data Flow

3.2 Research Questions and Scopes

This section states the three research questions and the scope of the study. The goal of these research questions is to surface the concrete rules, controls and audit needs that slow down or reshape the delivery. The questions also focus on listing out the practices that teams actually use and can repeat to satisfy regulatory requirements. And finally, the questions ask about the adaptive pathways linking a constraint to a practice and to the evidence which proves that the adaptive practice works. This whole research work will be followed by three research questions below:

RQ1: What specific constraints limit the implementation of Agile and DevOps practices in compliance sensitive industries?

RQ2: How do teams adapt their processes, tools and culture to meet both agility goals and regulatory demands?

RQ3: Which practical pathways successfully link regulatory constraints to the right practices and acceptable evidence?

The third research question will be answered with a framework of constraint-practice-evidence paths. The same coding scheme is used for interviews and case studies to allow direct comparison. Findings are reported with short examples and the minimal evidence that supports each path. The goal is to provide a set of results that teams and auditors can act on.

The scope focuses on regulated software delivery in banking (Finland, France, United States of America and Bangladesh). Comparative cases from healthcare and energy sectors are included to check transferability. The unit of analysis in this study is the delivery team and its CI/CD pipeline. Data is collected from semi-structured interviews and from selected case-based papers that report real practices. The study looks at the flow points such as planning, coding, building, testing, security scanning, approvals, release and operations. Artifacts in scope include backlog items, pull requests, test and scan reports, release notes, approval logs and artifact hashes or SBOMs. Evidence topics include traceability, segregation of duties, change control, privacy-by-design and tool or pipeline trust. The study does not attempt a full legal analysis of standards or a full tool-qualification program. AI-specific regulation is noted as future work and is outside the current scope. Claims are limited to patterns that appear in more than one source and are described in clear and practical terms.

3.3 Interview Study

This study includes an interview phase to understand the real delivery work under regulation. Semi-structured interviews are used to gather examples, constraints, fixes and the evidence the teams keep. Interviews are read together with case studies to see what patterns repeat and what differs by sector. Notes and recordings have been transcribed, anonymized and coded with the same scheme as the cases. The findings are reported in simple terms with short examples and clear links to evidence.

3.3.1 Participants and Sampling

Five interviews were conducted virtually via Google Meet. Two Participants were Software Development Team Leads from two leading banks in Finland. One participant was a Blockchain Software Development & DevOps Team Lead from the energy sector in France. One participant was a Software & DevOps Lead from the financial sector in United States of America. One participant was a CISA auditor who inspects banking software and infrastructure in the leading banks of Bangladesh. Roles were chosen to cover delivery, operations and assurance views. Sectors were chosen to reflect banking and financial sector as the main setting and energy as a useful comparison. Purposive sampling was used to reach people who work with Agile and DevOps under strict rules. Snowballing was then used to add contacts where needed. Typical experience covered team leadership, pipeline work and audit preparation. Sessions were scheduled for 35 to 40 minutes. Conversations were held online for convenience and recorded with the consent of the participant. Transcripts were created and checked for accuracy. Organization and personal names were removed or masked. Each participant was assigned a code for analysis. No sensitive customer data was requested or collected. Participation was voluntary and could be ended at any time. The interview set is small, so findings focus on clear patterns that appear across more than one source. Case studies are used to strengthen and compare the interview patterns. The invitations explained the study aim, privacy steps and time needed. Eligibility required current or recent responsibility for regulated Agile-DevOps delivery or audit of such teams. All recordings and transcripts were stored in a secure folder with restricted access. Table 3.3 below shows the information about the interviewed professionals which includes the geographic region, industry and company position of each.

Interview ID	Sector	Designation	Country	Region
INT-01	Banking	IT Manager	Finland	Europe
INT-02	Energy	Senior DevOps Engineer	France	Europe
INT-03	Financial Services	Senior Software Engineer	United States of America	North America
INT-04	Audit and Assurance	CISA Auditor	Bangladesh	Asia
INT-05	Banking	Lead Developer	Finland	Europe

Table 1: Interview Participants' Industries, Positions and Regions

3.3.2 Interview Guide

The interviews started with the introduction, purpose and scope of the study. A short interview guide was used to keep a steady focus and which could still allow open answers. The guide asked which regulations affect Agile and DevOps work. The guide then asked how those rules change CI/CD practices. A request for a real conflict example was included to ground the answers. A set of questions were asked to know which processes were changed to balance speed and compliance. A follow-up question asked how teams maintain that balance in daily work. The participants were asked about tools that support balance. Release timing and sprint planning were covered considering approval steps and change windows. Common mistakes and tips for other teams were also asked at the end. Each question was followed up with asking to know the point of view of the participant and any relevant examples they could share. The order moved from general to specific to make the talk easy to start. Questions were refined after the first interview to probe unique practices. Neutral wording was used to avoid leading answers. The prompts were kept short and asked for concrete cases. The clarifying questions were used only to reach clear examples or evidence. Industry terms were explained in simple language wherever it was needed. Answers were encouraged to point artifacts such as tickets, pull requests, test or scan reports and approval logs. The interviewer remained neutral and did not judge practices. Each interview ended with a short recap of key points to confirm the understanding. The guide was checked for clarity and flow before use. Minor wording changes were made to improve readability. The same opening script explained scope, consent and anonymization. The same guide was used for all five interviews to support comparison. Field notes were taken during and after each session to capture examples and wording that would help with the coding. Definitions of key terms like traceability, evidence pack, SoD etc. were stated at the beginning of the interviews to keep a common frame for all participants.

3.3.3 Data Collection Procedure

The interview data were collected using a simple and repeatable process. Semi-structured interviews were chosen to let the participants explain real and give examples. Participants were identified and recruited based on a clear criteria. The recruited participants are IT Professionals who have hands-on experience with Agile and DevOps in regulated settings, a current or recent role in delivery or audit and direct exposure to change control, security or compliance reviews. Purposive sampling was used to reach these roles. Most outreach was done through LinkedIn. A few contacts were reached through personal networks. Each invitation explained the study aim, the time needed, privacy steps and that participation was voluntary. Consent was confirmed before starting any recording during the meeting. Interviews followed a stable format. Sessions were semi-structured, lasted 30-60 minutes and were held online. Google Meet was used for most calls. Audio was recorded with permission. The guide was aligned to the research questions. Questions on RQ1 asked which rules and controls limit Agile and DevOps and where the flow breaks. Questions on RQ2 asked what teams changed in process, tools or roles to meet both speed and compliance. Open prompts were included to surface early ideas for RQ3, such as which evidence convinced auditors that a practice worked. The questions asking order moved from general to specific. Neutral wording was used during the interview. Clarifying questions were used only to get concrete examples and names of artifacts.

The timeline of the interviews spanned from mid-August to mid-October. Five interviews were completed in this window. Most sessions were scheduled in the late afternoon to suit the work hours of the participants. Scheduling these interviews was a key challenge. It took time to find the right professionals on LinkedIn and to receive consent to join an interview. One bank lead had limited schedules available due to release work and this was resolved by offering flexible slots and a short follow-up. Another participant requested anonymization details before confirming. A summary of data handling and storage was sent to address this. No in-person meetings were conducted. All files were named with codes, stored in a secure folder and checked for accuracy after transcription. The sample was limited to five interviews across banking and energy. Although it was planned to gather data from the healthcare sector as well, any interviews could not be secured within this timeframe. This gap is noted as a limitation of the study. For the overall study, cross-sector coverage is strengthened by triangulating interview themes with documented cases from prior studies.

3.3.4 Ethics, Consent and Anonymity

Ethical conduct was ensured at every step of the interviews. Before any interview, an information note was shared with the participants that explained the study aim, the time needed, data use, privacy steps and contact details. Informed consent was obtained for every session and verbal consent was recorded at the start of the call when written consent was not practical. The participation was voluntary. Participants were free to skip any question or stop the interview at any time without giving a reason. Anonymity was protected by removing the names of people and organizations from notes and transcripts. Role-based identifiers and neutral codes were used in all records and reporting. No internal documents, screenshots or customer data were collected. Any examples discussed, were described in general terms only.

Data were stored securely. Audio files and transcripts were saved in an encrypted and password protected folder with access limited to the researcher. Filenames used only the anonymous codes. Consent emails and consent notes were stored separately from the interview data. Transcripts were checked against the audio and then used for analysis. Audio files will be deleted after transcription checks and completion of the thesis. No files were shared outside the research workspace and no cloud links were made public.

These steps ensured that interviews were conducted with respect for privacy and autonomy, data were handled with care and the rights of participants including the right to withdraw, were maintained.

3.3.5 Data Handling and Management

Data were handled with a simple, secure and traceable process. All the files were stored in an encrypted and password-protected folder. Access was limited to the research workspace only. Filenames used anonymous codes (for example, INT-01) and no personal or company names were mentioned there. Audio was recorded with consent and saved immediately to the secure folder. Transcripts were produced, checked against the audio and lightly corrected for obvious errors. Names of people, teams and organizations were removed or masked during the transcription process. Any sensitive operational detail that could identify a person or any company was generalized. Consent emails and consent notes were stored in a separate subfolder and not mixed with interview content.

Data were organized by a clear folder structure like audio, transcripts, codebook, coded-data, figures and consent. Version control was maintained for the codebook and the coded dataset. Each update was saved as a new version with date and a short note in a simple changelog. Basic integrity checks were applied by reviewing file sizes, dates. The files were again read after moves or backups.

Analysis files were kept in open and portable formats wherever possible in formats like .txt, .csv and .docx. Coded segments were stored in a single worksheet to keep the audit trail simple. The coding scheme was documented in a small data dictionary. Any data exports used to create figures were saved with the exact version tag of the source sheet. Backups were made on a regular schedule to a second encrypted location. No files were sent by public links and no cloud sharing outside the private research storage was used. If a participant requested removal, all linked files (audio, transcript rows, codes) would be deleted from primary and backup locations.

The retention will follow a short and stated timeline. Audio files will be deleted after transcript checks and after the thesis is completed. Anonymized transcripts, the codebook and the coded dataset will be retained only as long as needed for any examination and any required revisions. After that point, remaining materials will be archived securely or deleted according to the institutional guidance. These steps ensure confidentiality and minimize the risk. It also keeps a clear audit trail from raw audio to coded results.

3.4 Case Study Review (Secondary Data)

This section reviews case studies used as secondary data to complement the interviews. The aim of including case studies is to capture the real-world situations where Agile and DevOps met regulatory needs, what blocked the project progress, what was changed and what evidence proved control. Cases were read in the same way as the interviews and coded with the same scheme. Examples include a large bank DevOps adoption, COBIT and Scrum alignment, medical device delivery under IEC 62304, ISO 9001 in Scrum, GDPR in SMEs and other sector cases with concrete practices and metrics.

3.4.1 Case Selection Criteria

Cases were selected based on the scenarios they described. The specific scenarios with Agile or DevOps under explicit regulatory or governance constraints, were included. Each case needed to show the concrete challenges like SoD, change windows, traceability, privacy and named adaptations like release gates, QA sign-off, IaC, privacy gates etc. Preference was given to sources that reported artifacts or metrics like coverage gates, scan pass rates, cycle time impact etc. rather than describing only the concepts. Multi-sector coverage was targeted across banking, healthcare or medical devices, energy and telecom to evaluate the transferability of Agile-DevOps compliance strategies. Currency and clarity were considered. Recent peer-reviewed papers and well-documented industry reports were prioritized. Cases that lacked methods, evidence or any clear context were excluded or flagged as low confidence. Table 2 below gives a quick overview of the selected case studies and their main focus.

Case ID	Case Focus
CASE-01	CI/CD in financial sector with strict SoD
CASE-02	COBIT-Scrum
CASE-03	Medical device (IEC 62304)
CASE-04	Medical DevOps (ISO 13485)
CASE-05	ISO 9001 AND Scrum
CASE-06	ISO 9001 documentation
CASE-07	Standards and Agile Value
CASE-08	GDPR in SMEs
CASE-09	Fintech Regulatory Crisis Case (Paytm)
CASE-10	Problem based Agile regulatory policy
CASE-11	IEC 62443-4-1 in DevOps
CASE-12	GDPR at scale (Ericsson)
CASE-13	Industrial S2C-SAFe framework for IEC 62443 (Siemens)

Table 2: Overview of Thirteen Case Studies Used as Secondary Data

3.4.2 Sources and Search Approach

Case studies were sourced and searched through LUT Primo[1] by using terms such as “DevOps regulated”, “COBIT Scrum”, “IEC 62304 Agile”, “ISO 9001 Scrum” and “GDPR DevOps”. Filters were set for English language and software or IS venues. Reference lists in relevant papers were used for snowballing. Only items with enough detail to identify constraints, practices and evidence were kept. PDF files were downloaded and stored with clear names and sector tags. A short log was kept for each case such as venue, year, sector, regulation focus and a one-line rationale. The final set includes the large-bank DevOps case, COBIT–Scrum in a Tier-1 bank, MED-EL and other medical-device cases, ISO 9001 in Scrum, GDPR in SMEs and related industry reports with measurable outcomes.

3.4.3 Data Extraction Template

A single data extraction template has been used so that cases can be easily compared with each other. Each entry in the template starts with Case ID, year, venue or type, sector and organization context (size, product, delivery model). IEC 62304, ISO 9001, COBIT, GDPR etc., whichever regulation or framework is in focus in the cases, has been noted. The delivery setup is described in plain terms like Scrum or Kanban, CI/CD tools, environment model etc. A short scenario line explains what was attempted and why regulation mattered. For RQ1, the constraints are captured in simple labels like release completeness, SoD, traceability, privacy, tool trust etc. For RQ2, the practices are recorded as actions in the workflow such as release gate with policy-as-code, IaC with signed artifact promotion, privacy gate etc. For RQ3, the evidence of proving control is listed such as tests and scan reports with tool or policy versions, approver and timestamp, release notes, artifact hash or SBOM, QA sign-off, DPIA status, masking or cleanup logs etc.

[1] LUT Primo is the official discovery and search system of Lappeenranta-Lahti University of Technology (LUT). It provides unified access to the university library’s electronic databases (IEEE Xplore, ACM Digital Library, ScienceDirect, SpringerLink etc.), e-journals, e-books and the LUTPub repository containing all LUT master’s and doctoral theses.

All metrics are copied with units. Artifacts are named like ticket types, PRs, CI logs, gate logs and how they link to releases is noted. Tooling and versions are recorded if they have been stated. They have been recorded with notes on pinning or qualification. Outcome statements are kept brief like faster releases, fewer errors, audit time reduced etc. Quotes or paraphrases that illustrate a point are saved as short lines. Limits and bias are captured. Each case ends with a one-line constraint, practice and evidence pathway written in compact labels so that it can feed the pathway map. This template aligns directly to RQ1, RQ2 & RQ3 and supports the later C-P-E integration framework.

3.4 4 Case Quality Check

Each case studies were screened with a short quality checklist. The context of the cases had to be clear specifying the sector, team type or regulation. The methods like interviews, workshops, action research, retrospective etc. used in the case studies had to be stated or at least implied with enough detail to judge the credibility. The specificity of the evidence was checked with named artifacts, logs or metrics. These were preferred over general claims. The words suggesting cause and effect were reviewed. Only claims that were clearly supported by the data were kept. The possible conflicts of interest were noted when cases were written by the organization involved. Date and currency were considered where rules or tools change fast. Sector fit and regulation fit were judged against the thesis scope. The cases that aligned with multiple other sources on the same point, were marked as confirmed. Unique points were kept, but those were labelled as low confidence unless supported elsewhere by other evidence. Where the sampling or method was weak, the case was used only to illustrate a pattern already seen in stronger sources. Inclusion and confidence decisions were recorded in the case log. Labels in the extraction were kept stable to avoid drift. When uncertainty remained, the constraint, practice and evidence pathway from that case was included with lower weight in the later analysis. This approach keeps the review transparent and balanced. It ensures that the pathways reported in results are grounded in cases with clear context and observable evidence like the bank pipeline case, COBIT-Scrum alignment, medical-device flows, ISO 9001 in Scrum, GDPR changes etc.

3.5 Data Analysis

This section explains all the processes of coding, organizing and interpreting the qualitative data to answer the three research questions. Both interviews and case studies were analysed with the same rule of identifying constraint, practice and evidence so that results can be directly compared. The analysis was done following four main stages.

In the Transcription and Preparation stage, every interview was transcribed, anonymized and stored securely.

Then in the Open Coding and Categorization stage, the limits, actions and proof were tagged as constraint, practice and evidence.

After that, in the Pattern Grouping stage, similar codes were grouped into broader themes such as security compliance, tool restrictions or regulatory burden.

Finally, in the Cross Source Triangulation stage, findings from interviews were checked against the 13 case studies to confirm repetition or difference across sectors.

Figure 2: Coding Flow

3.5.1 Coding Approach for Interviews

Each interview script was read line by line so that any core points or insights would not be missed. Segments were highlighted whenever a participant mentioned a regulation, a workaround or any kind of proof. Initial codes were then merged under main themes that were recurring in more than one interview (Table 3). Theme IDs with C denote the constraints or limitations and IDs with P denote the practices adapted by teams. Table 3 highlights the constraints and practices that were observed in the interview scripts and case study data.

Theme ID	Description	Indicator	Frequency (across 5 Interviews)
C1	Regulatory approval and change management delays	"Every production change needs CAB approval"	5/5
C2	Data protection and privacy rules	"Must get sign-off from data verification group before deploying"	4/5
C3	Tool and environment restrictions	"Postman was banned for storing scripts in cloud"	3/5
C4	Security and vulnerability scanning	"MEND blocks pipeline if vulnerability is found"	5/5
C5	Segregation of duties and access control	"No developer can touch production directly"	5/5
P1	Gated CI/CD pipelines	"CAB review stage in Jenkins pipeline"	4/5
P2	Policy-as-Code and automation	"OpenPolicyAgent(OPA) blocks non-compliant containers automatically"	3/5
P3	Compliance ownership to lawyers and platform engineers	"Lawyers update policy files, pipeline enforces it"	2/5
P4	Continuous and parallel documentation	"Strict rules to document process and external access"	3/5

Table 3: Interview Coding Themes

After coding, constraint, practice and evidence triplets were created for each unique situation. These triplets were later used to answer each research question.

3.5.2 Process of Cross-Case Synthesis

Each of the 13 case studies was studied to identify if the interview patterns were unique or common. The cases represented banking, healthcare, telecom, energy and fintech sectors. After the individual coding of the interviews and case studies was completed, a cross-case synthesis was performed to identify the patterns that are observed across different sectors. The process of cross-case synthesis is shown in Figure 3 below.

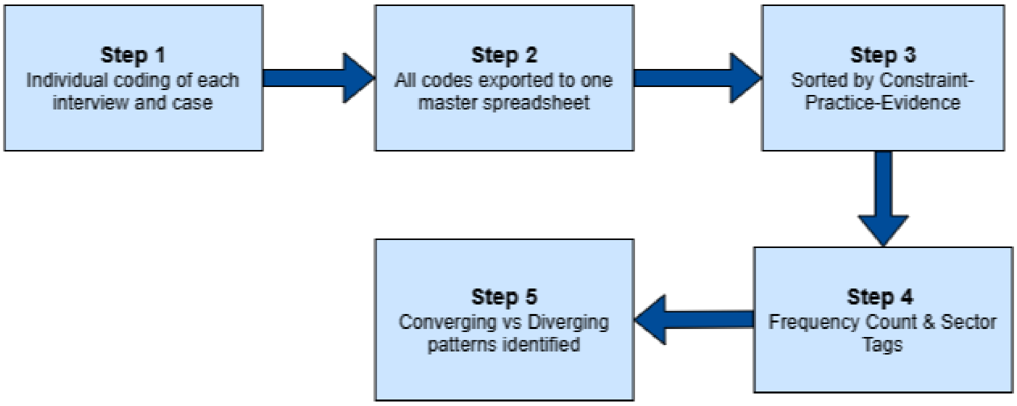


Figure 3: Cross-case Synthesis Process

The results of this synthesis are presented in Chapter 4.

3.5.3 Integrating Interviews and Cases

After analysing the interviews and case studies individually, both datasets were merged. Every interview pattern was checked against the case study table. The integration process linked the practical realities described by interview participants with the documented experiences found in the case studies. This step made the credibility of the results stronger and ensured that findings were not limited to one company or context. All coded elements from the interviews were compared with similar patterns emerging from the case dataset. Where both sources reflected the same phenomenon, the theme was marked as convergent. Themes that appeared in only one source, were kept as context specific. This helped to explain the variations between different sectors. The comparison provided a clear consistency between practitioner experience and published evidence. In both interviews and case study data, it was noticed that organisations faced similar forms of regulatory constraints. Governance approvals, segregation of duties, privacy rules, audit trail requirements etc. are some common constraints found. Also, adaptive practices like automation, policy as code and continuous documentation were adopted to maintain the agility. Finally, both sources confirmed a shared shift towards data-driven evidence. This practically means that the same constraint, practice and evidence logic holds across diverse sectors. Therefore, this integration strengthens the claim that agility and compliance can coexist when governance becomes a built-in technical function rather than being a separate managerial layer.

The idea of creating the C-P-E framework emerged while reading the first two interview scripts and the first three published case studies. Every participant in the interviews and almost every case first described a strict rule that affected the agility of regular DevOps operations, then explained the new tool or process their team built to solve the issue and finally showed which type of proof the auditors usually accepted. This same order of problem, solution and proof appeared repeatedly in completely different organizations and sectors. As this pattern was so clear and repeated so often, I decided to organize all the data using these three simple categories of constraint, practice and evidence. The C-P-E model presented in Chapter 4 is a direct outcome of this data-driven process. Figure 4 below represents how interview data and case study evidence converge into a three-layer integration model. The first two boxes represent the two data sources providing input for an integrated analysis. The middle layer shows three recurring dimensions that connect both sources with shared constraints, adaptive practices and evidence mechanisms. Together, these form a unified Constraint, Practice and Evidence (C-P-E) framework, which demonstrates how regulatory compliance and agility can be achieved simultaneously through automation and continuous documentation.

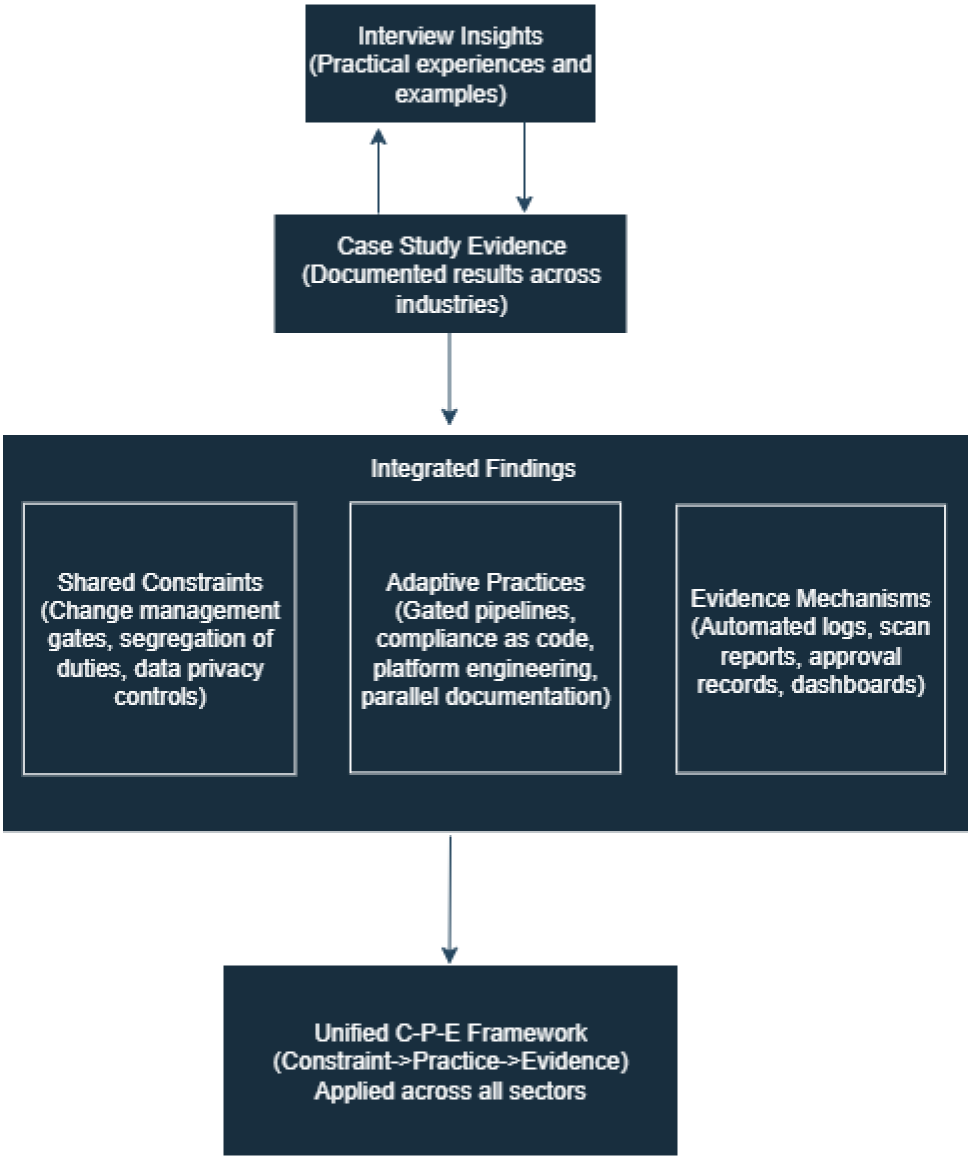


Figure 4: Three-layer Integration Model

3.5.4 Tools Used

Throughout the research process, a set of accessible and reliable tools was used to manage data systematically and maintain the transparency which is shown in Table 4 below. Preference was given to simple and traceable tools rather than relying on advanced qualitative software so that the process remains open, replicable and easy to audit.

Purpose	Tool	Application
Interview Transcription	Google Meet Recording, Tactiq and Microsoft Word	Manually transcribing meeting from the recording and AI based transcription for cross-checking
Data Coding and Categorization	Microsoft Excel	Applying the constraint, practice, evidence (CPE) codes, sorting and tagging flexibly
Cross-Case Comparison	Excel Pivot Tables	Grouping and counting recurring patterns across all sectors
Data Visualization	PowerPoint	Creating diagrams and flow charts
Reference Management	Zotero	Storing and formatting interview transcripts, academic sources and case study references
Document Preparation	Microsoft Words	Drafting all sections and embedding tables and figures.

Table 4: The Tools Used for Data Management

3.6 Limitations of the Method

The research used triangulation and careful coding to ensure validity, but there are some methodological limitations in the process. Recognizing these limitations and boundaries helps to interpret the results with appropriate caution and provides guidance for future work. The first limitation was the limited number of interviews. Only five professionals were interviewed who are primarily from the banking and energy sectors. This provides depth but limits the generalization to other regulated domains such as healthcare or government IT. However, the thirteen case studies broadened the scope and compensated for sectoral imbalance. Then comes contextual bias. Interview responses are based on individual perceptions which may reflect personal or organizational bias. Participants might emphasize success stories or downplay internal challenges. The triangulation process with published case studies reduced this risk although subjective interpretation cannot be entirely removed. Another limitation arises from the case studies as they differed in detail and evidence. Some cases provided full metrics and some only offered qualitative insights. This unevenness may slightly affect the accuracy of the cross-case comparison. But the coding framework helped to maintain consistency. In addition to these, there was a single researcher’s interpretation in the whole data analysis process. The consistency was maintained through a strict CPE codebook and repeated review. But having multiple coders could have further strengthened the reliability. Then comes the evolving regulatory environment. Regulations like the EU’s Digital Operational Resilience Act (DORA) and GDPR continue to evolve. The practices identified in this study represent a snapshot of compliance methods between 2024 and 2025 and they may need revision as rules are updated. Finally, there was limited quantitative validation since the study focused on qualitative synthesis rather than numerical measurement. Future research could add a quantitative evaluation to complement the qualitative insights.

Despite having these limitations, the combined use of multiple data sources, transparent coding and clear documentation makes the credibility of this research stronger. The triangulated design with interviews and case studies ensures that the findings reflect both practitioner experience and documented industrial experience. This offers a reliable foundation for the following Results chapter.

3.7 Chapter Summary

This chapter explained the process of data preparation and analysis. A single coding frame with constraint, practice and evidence was used for all materials. Interview data provided direct practitioner experience while the case studies provided wider sectoral confirmation. The cross-case synthesis connected the recurring patterns and separated strong, moderate and unique themes. The outcome of this analysis is a structured foundation for the next chapter. The detailed result in the next chapter shows what constraints limit the Agile-DevOps practices, how teams adapt and what evidence proves control.

4. Results

This chapter presents the findings from the thematic analysis of the five semi-structured interviews and thirteen documented case studies. The results are organized according to the research framework of Constraint, Practice, Evidence and aligned with the three research questions:

RQ1: What specific regulatory constraints limit the implementation of Agile and DevOps practices in compliance sensitive industries?

RQ2: How do development teams adapt their processes, tools and culture to meet both agility goals and regulatory demands?

RQ3: Which practical pathways successfully link regulatory constraints to the right practices and acceptable evidence?

The analysis integrates both primary (interview) and secondary (case) datasets through a systematic coding and categorization process. Each dataset was examined to identify the mostly mentioned regulatory constraints, adaptive practices and compliance evidence mechanisms. The aim of this chapter is interpreting the dominant, recurring and contrasting themes that appeared across the dataset. Each section provides a narrative description of the patterns and relationships observed in the data and these are supported by summarized tables and conceptual figures.

4.1 Data Overview

The dataset consisted of eighteen sources and there were five interviews (INT-01 to INT-05) and thirteen case studies (CASE-01 to CASE-13). Each source was coded according to the C-P-E framework, resulting in 65 coded records in total. The record here in this research means a coded piece of data which is either a sentence or a short paragraph. Every record contains a regulatory constraint, the adaptive practice and the evidence. All coded entries were assigned one of nine constraint categories, nine practice categories and eight evidence categories as identified in the codebook. The data source summary is shown below in Table 5.

Data Source	Type	Count of Records
Interviews	Primary (INT-01 to INT-05)	26
Case Studies	Secondary (CASE-01 to CASE-13)	39
Total		65

Table 5: Data Source Summary

The interviews offered detailed practitioner insights and the case studies provided documented examples of process adaptation and evidence mechanisms. Together, these datasets produced a rich and triangulated view of how Agile and DevOps principles are negotiated under regulatory control. Each transcript and case description was examined line by line to identify the constraints, adaptive practices and evidence mechanisms. These codes were then grouped into higher order themes. The following sections describe these major themes and how they interact with each other.

4.2 Cross-case Synthesis

Up to 3 constraint, practice and evidence records were found in each case. The goal of cross-case synthesis was for comparing instead of counting. Each theme found in the interviews was checked. If one theme also appeared in multiple cases, then it was classified as a strong pattern. If it appeared in one case only, then it was kept as a context specific insight. The key patterns across sectors are shown below in Table 6.

Sector	Main Regulatory Constraint (RQ1)	Adaptive Practice (RQ2)	Evidence Artifact (RQ3)
Banking and Finance	Rigid change management and segregation of duties (no developer access to production)	Introduced gated CI/CD pipelines and policy-as-code to automate the approvals and access reviews	CAB approval tickets, immutable log exports and ELK dashboards showing change history
Energy	External partner integrations required multiple logins and GDPR/MiCA compliance checks	Created continuous documentation and periodic access reviews, added policy enforcement checks for every new tool	Access review reports, policy audit logs, documented proof-of-concepts for third party tools

Healthcare	Heavy safety and verification documentation required by ISO 13485/IEC 62304 standards	Embedded a quality interface layer within DevOps pipelines to connect automated testing with compliance review	Verification and validation reports, signed test artifacts, automated traceability reports
Telecom (Large scale agile)	Distributed global teams struggled with inconsistent GDPR interpretation and late compliance reviews	Implemented shared compliance wikis and automated tracking tools to ensure consistent understanding and synchronized timelines across all teams	Wiki based change records, compliance dashboards and audit ready reports generated automatically
Industrial and Manufacturing (Security focused)	Security standards like IEC 624434-1 slowed down continuous delivery due to manual reviews	Used automated SAST/DAST scanning and security champions within teams to ensure compliance without delay	Pipeline security reports, automated vulnerability dashboards and audit snapshots of each release
Cross-sector Observation	Common challenge is balancing audit traceability with delivery speed	Core solution is implementing compliance as code which will embed all checks and policies directly into CI/CD pipelines	Automated logs and approval reports accepted by auditors as official compliance evidence

Table 6: Cross-case Synthesis

Across all 13 cases, three clear trends were observed.

Firstly, Compliance embedded in automation. Instead of treating regulation as a separate stage, organizations encoded rules directly into their DevOps timelines. This compliance as code approach was the most effective and repeatable solution across sectors.

Secondly, Parallel documentation. Teams maintained continuous documentation rather than making separate documents and reports at the end. These documents were often directly linked to their sprint or CI/CD tools which ensured every change was traceable without slowing down the development.

Thirdly, Shifting from manual to verifiable evidence. Auditors increasingly accepted the automatically generated pipeline logs, risk dashboards and verification reports as valid proof of compliance. This reduced audit preparation times by 50-70% in several banking and industrial cases.

The cross-case relationship between constraint, practice and evidence has been represented in Figure 5 below. This map visualizes the strongest recurring patterns from the eighteen data sources and shows how the common constraints directly lead to specific practices which also generate evidence. This map also confirms the universality of the resulting C-P-E framework.

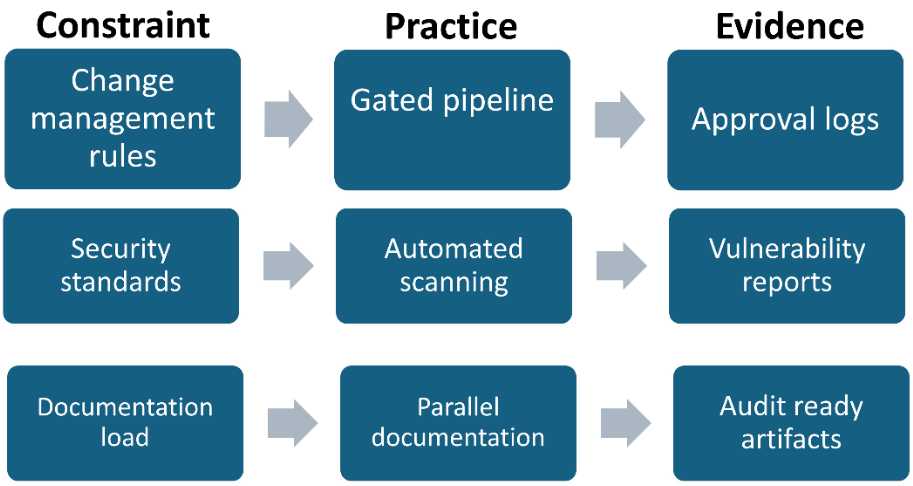


Figure 5: Cross-case Relationship Map

4.3 Constraint Analysis

The first stage of analysis addressed RQ1 and explored the regulatory and organisational constraints that limit the implementation of Agile and DevOps practices in compliance sensitive environments. The thematic analysis revealed eleven main constraint themes that were repeatedly emphasized across both interviews and case studies.

Mostly Observed Constraint Categories	**Access Control and SoD**
	Delay in Workflow
	Organisational Regulatory Burden
	Restricted Development Environment

Table 7: Mostly Observed Constraint Categories

These recurring patterns indicate that regulatory control is deeply embedded within technical and procedural layers of software delivery. Security, access and segregation of duties (SoD) restrictions were present in every organization studied, suggesting that governance and risk management processes are inseparable from delivery workflows. Participants also acknowledged that these controls create friction between operational speed and regulatory assurance. These controls often lead to delayed workflows and cause regulatory burden.

These results indicate that excessive restrictions, delayed workflow and access control obligations are the most widespread inhibitors to agility in regulated industries. The patterns are consistent across both datasets. All five interview participants described that segregation of duties (SoD) and multi-layer approval processes are mandatory, but these make the working process very slow and often create a burden. The majority of the case studies reflected similar security-driven gating mechanisms.

In summary, RQ1 is answered by demonstrating that constraints are both regulatory (GDPR, DORA, IEC Standards) and procedural (CAB approvals, access policies). The core challenges faced by the teams represent system barriers rather than isolated process inefficiencies.

4.3 Summary of Major Constraint Themes

A detailed review of the mostly mentioned constraint themes shows the major challenges the studied organizations face because of regulations in Agile and DevOps workflow. The mostly observed constraint categories and the specific challenges faced by the studied organizations, are summarized below:

1. **Restricted Development Environment**
    a) Strict tool and data residency policies restrict the use of external and cloud-based platforms for development and testing.
    b) Mandatory legal and compliance reviews are required before each release in continuous integration and delivery.

c) Developers often require operating within pre-approved environments using authorised tools only with periodic access reviews to ensure compliance.
d) Testing is limited to synthetic and non-production data which restricts full validation and increases the risk of discrepancies with real world data behaviour.
e) Deployment activities are limited to approved business hour windows which prevents out-of-hour releases and reduces flexibility.
f) Multi-level formal code approval process at multiple stages hampers the continuity of development and release cycle.

2. Access Control and Segregation of Duties (SoD)

a) Strict separation of role and manual handover requirements between teams reduces deployment and delivery efficiency.
b) Integration with external systems requires complex authentication, multi-level approvals and restricted namespaces.
c) Deployment responsibilities are limited to operation teams while developers can only verify outcomes and cannot perform any operational tasks.
d) OTP based remote sessions and strict security controls frequently interrupt CI/CD processes and hinder smooth collaboration.
e) Periodic access reviews disrupt ongoing agile workflows and interrupt continuous integration and delivery cycles.

3. Delay in Workflow

a) Mid-sprint reviews and changes mandated by compliance teams extend the sprint duration and hinder the timely completion of planned stories.
b) Mandatory Change Advisory Board (CAB) approvals are required before every production deployment which causes time lags even for minor fixes and urgent updates.
c) A strict unit before integration testing policy delays continuous integration and slows overall CI/CD throughput.

d) Mandatory DPIAs, data masking and privacy by design reviews extend the release timelines and add additional validation steps before deployment.
e) Extensive manual security reviews are required before deployment which slows CI/CD pipelines by over 20% and delays delivery efficiency.
f) Lengthy compliance verification processes extend release timelines by up to six weeks which significantly delays delivery cycles.
g) Collaboration with external partners requires multiple authentication and approval steps which creates delays in integration, coordination and technical support.

4. Organisational Regulatory Burden

a) Extensive bureaucracy and multi-level approval chains within large firms slow the decision-making process and hinder timely regulatory compliance.
b) Regulation enforces 100% test coverage requirements which increase testing workload and extend release preparation time.
c) Developers are directly responsible for extensive compliance documentation and validation tasks which diverts focus from core development activities.
d) Outdated legacy systems and limited DevOps expertise in regulated industries restrict the team’s effectiveness on modern automation and continuous delivery practices.
e) Traditional governance rules and steering committee controls often conflict with Scrum role authority which particularly diminishes the decision-making autonomy of Product Owners.
f) Compliance with the COBIT framework requires an independent QA function which adds extra review layers and extends the overall release cycle.
g) Additional regulatory requirements create inconsistencies in the existing quality management system which make the agile processes partially unsupported by formal compliance structures.

h) Extensive verification, validation and risk documentation required by safety-class software, significantly increases the effort and duration of development cycles.
i) All project activities must be fully completed before release approval which creates clash with continuous delivery and the iterative flow of Agile and DevOps practices.
j) Strict adherence to ISO Clause 7 requires extensive documentation and record-keeping which places additional workload on teams and slows down Agile iterations.

These constraints are not any isolated issues. In fact, these are the interconnected elements of a broader governance framework that prioritises compliance over speed and adaptability. The recurring patterns observed across different organisations demonstrate that regulatory expectations shape every layer of the delivery pipeline from tool selection and data handling to sprint planning and release management. These mechanisms ensure accountability and risk control, but they collectively reduce flexibility and slow iterative progress. Consequently, the following section examines how teams respond to these challenges by developing adaptive practices and automation strategies that enable them to align Agile and DevOps principles with the strict requirements of regulated environments.

4.4 Adaptive Practices and Cultural Responses

The second stage of analysis addressed RQ2 and it focuses on how teams have adapted their tools, processes and culture to meet both regulatory and agility goals. The thematic analysis revealed three categories of adaptive practices that appeared consistently across the interviews and case studies. These categories represent mature and cross sector responses to the constraints described above. Table 8 below shows these mostly observed categories.

Most Observed Key Adaptations	**Automated Access Governance**
	Automated Governance Pipelines
	Customized Workflows

Table 8: Mostly Observed Adaptive Practice Categories

These identified industry practices category indicates that automation has become the principal mechanism for integrating regulatory control with continuous delivery. The teams encode the compliance rules into systems instead of considering compliance as a separate phase. The participants often referred it as self-enforcing pipelines in the interviews. The case studies also reported this shift as governance by design. There were also other categories of prominent practices which indicated that organizations are focusing on human collaboration and cultural alignment with automation. Parallel documentation, policy as code and cross-functional integration are among those categories that are also practiced in banking, healthcare and energy sectors.

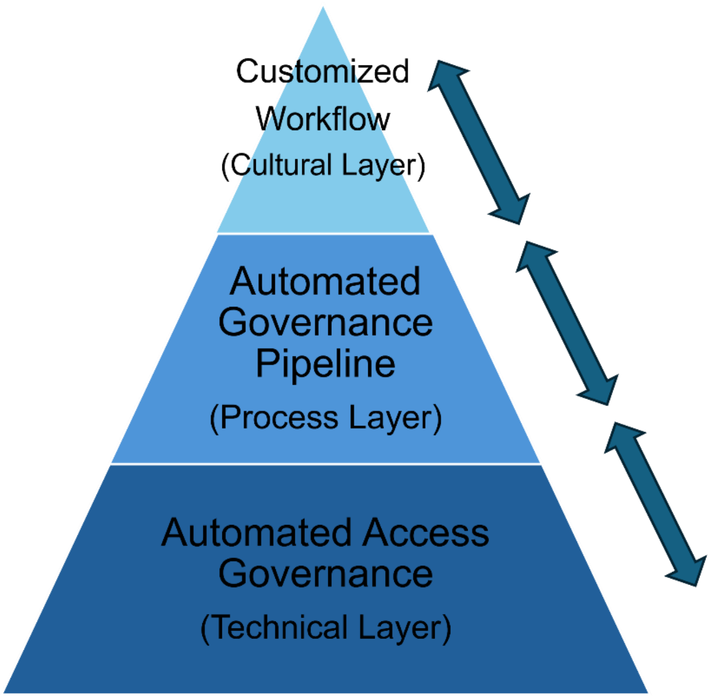


Figure 6: Hierarchical Representation of Adaptive Practices

The above hierarchical adaptation pattern in Figure 6 has been identified from the interview responses and study of the cases based on the adapted practices in the companies. Each layer in this representation reinforces compliance with the combination of automation and structured communication.

4.5 Summary of Key Adaptive Practices

Since the thematic analysis of the interview and case study data revealed several key practice categories reflecting the way organizations make compliance operational without compromising the delivery speed. These practices represent a blend of technical automation, process refinement and collaborative governance. The listed categories and their most significant adaptive practices are summarized below. Each of these practices addresses specific constraints and contributes to a balanced integration of Agile and DevOps principles within regulated environments.

1. **Automated Access Governance**
    a) Cloud based tools like Postman were discontinued under automated access control policies and they restrict external data storage and enforce compliance with internal security and residency rules.
    b) Periodic access reviews are enforced and local installations are limited to approved tools only. All software licenses are centrally managed to maintain controlled and compliant environments.
    c) Deployments are restricted outside business hours to minimize operational risk. Expert verification is required for any exceptional after-hour releases.
    d) Developers do not have any direct production access. Automated deployments are managed through Kubernetes service accounts and IAM role-based access control (RBAC) to ensure secure and compliant operations.
    e) The access of the developers is automatically limited to approved IDEs, toolsets and configurations only. License quotas and access permissions are enforced through automated governance controls.

f) User permissions in tools like JIRA, GitHub and other communication platforms are automatically reviewed to manage access levels and remove any inactive and outdated accounts.

2. Automated Governance Pipeline

a) Tools like SonarQube for code compliance and MEND for vulnerability scanning are integrated into CI/CD pipelines to automatically block deployments until all identified vulnerabilities are resolved.
b) Secure coding standards, SQL injection and output encoding are automatically enforced as architectural principles in all APIs to ensure consistent security compliance.
c) CI/CD pipeline are integrated with CAB and risk-based gating which makes the low-risk changes auto deploy after automated tests. High risk updates must pass through CAB review before release.
d) Compliance evidence is embedded within the CI/CD process where approval logs, vulnerability scan reports and immutable records are automatically generated as part of each pipeline run.
e) The ELK Stack is used for real time logging. Jira automatically assigns risk scores to streamline and automate continuous risk management within the delivery pipeline.
f) Automated compliance reporting significantly reduces audit preparation time and minimizes the need for manual verification across delivery cycles.
g) Compliance checkpoints are transformed into automated pipeline stages by integrating SAST as a pre-merge gate and executing CAB approvals as part of the pipeline workflow.
h) DORA driven controls and internal audit requirements are integrated directly into GitLab CI pipelines and Terraform modules and they ensure continuous and automated compliance enforcement.
i) A standardized CI/CD pipeline template was introduced with protected branches and service accounts to enforce compliant and auditable deployment practices.

j) Automated unit tests are configured as pre-merge gates which maintain the code quality and ensure compliance verification before integration.
k) Privacy checks like DPIA validation and masked data jobs are encoded into the CI/CD gates which ensure the automated enforcement of privacy by design requirements during each build.
l) Automated SAST and DAST scans are integrated into defined DevOps stages to enable continuous security validation throughout the delivery pipeline.
m) Automated tracking tools were implemented across distributed teams to enhance the visibility, synchronize the workflows and resolve the real time coordination related challenges.
n) Automated testing was integrated into SAFe pipelines to streamline the validation process and minimize the release delays in delivery environments.

3. Customized Workflow

a) Developers encode or mask data according to predefined company standards while handling unavoidable PII exposure. It assures legal and regulatory compliance.
b) Agile estimation methods are revised to include compliance related workload by allocating additional study points and creating dedicated sub-stories for compliance activities.
c) Compliance related tasks are incorporated into sprint planning and quarterly roadmaps which allows the teams to anticipate and prepare for regulatory approval milestones.
d) Instead of enforcing 100% test coverage, the teams mainly focus on 80% meaningful coverage through mutation testing. But they maintain full coverage for payment critical components.
e) Quality Management Procedures (QMS) are updated and the staff are trained to ensure traceable and auditable change management within Agile processes.

f) A lightweight Q-interface was introduced between DevOps and QMS functions to align continuous delivery practices with quality assurance requirements.

These workflow adaptations illustrate the strategic balance between agility and regulatory obligations. The organizations maintain delivery efficiency by integrating compliance tasks into sprint planning, refining the estimation methods and aligning the QMS processes with DevOps pipelines. And they are also meeting the strict governance standards. All these customized approaches enable continuous delivery to coexist with regulatory assurance which also reduces the clash between quality control and iterative development.

4.6 Evidence Mechanisms

At the third stage of analysis, the set of records used to evidence the compliance and audit requirements in Agile and DevOps delivery pipelines was examined to answer RQ3. The findings revealed a clear transition from manual record keeping to system generated and automated assurance mechanisms. The studied organizations mainly relied on continuously produced and traceable evidence that directly linked to technical workflows. These evidence mechanisms verify that the regulatory and security controls can be satisfied without interrupting the regular delivery workflow.

Across the studied organizations, three primary evidence clusters were consistently observed and those are summarized below:

1. Approval Logs, IAM Access Reports, CAB Approval Records

For ensuring traceable proof of deployment approvals and user access governance, these artefacts capture every authorization and access review within the CI/CD workflow.

2. Immutable System Logs, Security and Vulnerability Records, CAB Ticket Histories, DPIA Validation Records

Continuous logging and integration of automated scanning tools in DevOps pipelines generate immutable and timestamped evidence of system security testing, integrity and privacy by design enforcement.

3. Digital Sprint Artifacts, Compliance Checklists, Audit Dashboards, Automatically Generated Compliance Reports, Risk Scoring Summaries

The customized Agile and DevOps workflows produce continuous compliance visibility through real time dashboards and automated reports. Risk scoring summaries serve as audit ready evidence and these records confirm policy adherence and align with audit and risk management standards.

Altogether, these evidence mechanisms demonstrate the evolution of compliance verification into an embedded assurance process. These ensure the automatic record of required information for regulatory review without any kind of disruption in the delivery cycles.

4.7 Constraint-Practice-Evidence (C-P-E) Synthesis

The synthesis of the coded data from interviews and case studies revealed that the Agile and DevOps teams in regulated industries like banking, healthcare and energy, have established some structured delivery pathways that connect the mostly appeared constraints with adaptive practices and verifiable evidence. Each delivery pathway demonstrates how automation, procedural alignment and integrated reporting transform the regulatory barriers into auditable and repeatable processes. The thematic integration of constraint, practice and evidence relationships has been represented in the conceptual C-P-E framework shown in Figure 7 below.

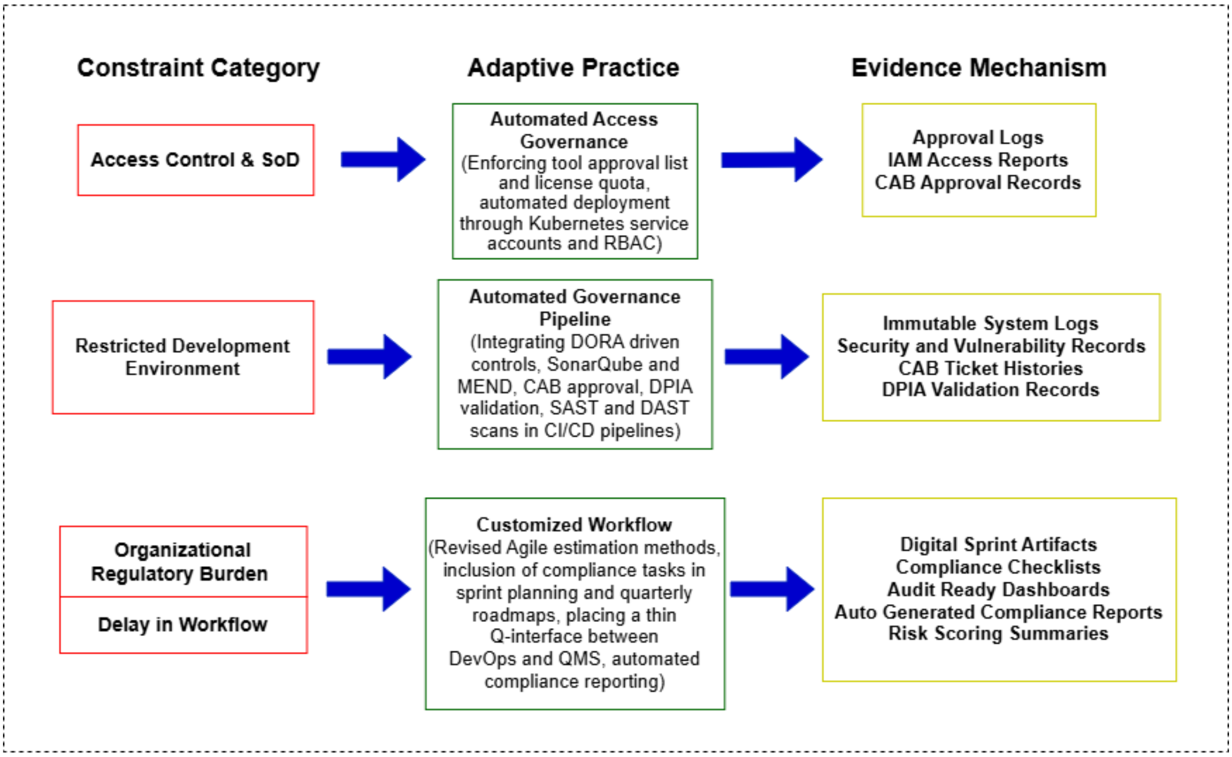


Figure 7: Constraint-Practice-Evidence (C-P-E) Framework

This thematic linkage directly answers RQ3 by showing that delivery pathways are formed when regulatory controls are translated into automated practices that produce verifiable evidence. This integration shows that compliance is no longer an external factor to development. In fact, compliance is an embedded function within software development. This resulting conceptual framework enables the teams to sustain agility by maintaining transparency, accountability and regulatory trust.

4.8 Chapter Summary

The results of the thematic analysis confirm that the regulated organisations often face persistent governance and security constraints which create several major challenges in Agile and DevOps adoption in the development lifecycle. However, the data also showed that the teams have developed several structured and sustainable adaptations to these constraints. Automation through access governance and governance pipelines has been observed as the unifying strategy that balances the delivery speed with regulatory assurance. Evidence creation has evolved from manual strict documentation to continuous, parallel and system generated documentation assurance. It reinforces compliance without sacrificing the agility. By synthesizing these patterns, the study demonstrates that compliance, agility, continuous integration and deployment can coexist when governance is embedded as an integral component of the development process.

This chapter has presented these findings through thematic interpretation and it leads to the discussion section in the next chapter where these results will be discussed within the broader literature and theoretical context.

5. Discussion

This chapter interprets and contextualizes the results presented in Chapter 4. It connects the thematic findings to the existing knowledge, elaborates on their theoretical and practical implications and identifies the opportunities for future development. The discussion session is structured around the three research questions and extends into sectoral and regulatory considerations. Finally, it introduces the emerging relevance of RegOps and MLOps as promising directions for future research aligning the significantly evolving artificial intelligence (AI) and machine learning (ML) systems with regulatory controls and expectations.

5.1 Brief Interpretation of Results

The study demonstrates that highly regulated organizations like banking, healthcare and energy can successfully align Agile and DevOps operations with compliance frameworks by applying automation, procedural alignment and data-driven assurance. The analysis revealed that the most significant constraints like restricted development environment, access controls and SoD, workflow delays and organizational regulatory burdens, are deeply embedded in organizational structure and technical governance.

In response to these key constraints, the organizations have introduced a number of adaptive mechanisms that transform the strict governance into executable automation. These adaptive practices include Automated Access Governance, Automated Governance Pipelines and Customized Workflows. The resulting integration creates a governance by design model where compliance is embedded as a technical capability within CI/CD pipelines rather than being a separate managerial operation.

Evidence generation has also evolved from static documentation to continuous and system generated artefacts. Audit trails, approval records and compliance dashboards now show the live visibility of verifiable artefacts during delivery. This demonstrates a clear cultural and technical shift towards a continuous assurance state where compliance proof is created automatically as a part of the development and release activities.

5.2 Answers to the Research Questions

5.2.1 Constraints in Regulated Delivery (RQ1)

RQ1: What specific constraints limit the implementation of Agile and DevOps practices in compliance sensitive industries?

The findings regarding this research question highlight four major constraint categories: Restricted Development Environment, Access Control and SoD, Delay in Workflow and Organizational Regulatory Burden. These constraints demonstrate how regulatory controls govern from tool selection to deployment in every stage of the delivery process.

Strict tool restrictions and data-residency policies prevent the teams from using modern cloud-based systems. Segregation of duties (SoD) and complex access controls reduce the autonomy and flexibility. Change Advisory Board (CAB) dependencies, extensive documentation requirements and multilevel approvals often create bottlenecks that result in slowing down the iterative delivery process.

However, these limitations also ensure the integrity, traceability and accountability in regulated sectors. The actual challenge lies in balancing control with responsiveness. The results confirm that these barriers are systematic and procedural and these must be managed through automation and workflow integration rather than avoiding and removing them.

5.2.2 Adaptations and Working Practices (RQ2)

RQ2: How do teams adapt their processes, tools and culture to meet both agility goals and regulatory demands?

The research identified a set of adaptive practices that make compliance operational through automation and structured workflows. Automated Access Governance limits tool usage and enforces policy compliance through access controls and role-based permissions. Automated Governance Pipelines integrate risk management, testing and approval gates directly into CI/CD workflows to meet the regulatory demands without affecting the iterative workflow. And the Customized Workflow integrates compliance planning into sprint cycles and QMS interfaces.

These types of adaptations demonstrate a shift from procedural enforcement to technical governance. By automating policy enforcement and reporting, the organizations have achieved regulatory compliance without affecting the delivery. The results also show the cultural adaptation where teams have accepted compliance as part of their engineering responsibility rather than treating it as an external oversight. This approach aligns with contemporary literature on Compliance as code (Rahman et al., 2019) and extends it by evidencing its implementation in multiple regulated sectors.

Ultimately, these adaptations convert the governance principles into coded automation which supports both continuous deployment and audit readiness.

5.2.3 Evidence Generation Pathways (RQ3)

RQ3: Which practical pathways successfully link regulatory constraints to the right practices and acceptable evidence?

The integration of constraints, practices and evidence mechanisms forms the Constraint-Practice-Evidence (C-P-E) framework. This conceptual framework illustrates the way automation bridges regulatory controls and verifiable outcomes. Each constraint category maps to some adaptive practices that automatically produce its own assurance artefacts. For example, the Restricted Development Environment category leads to Automated Governance Pipelines that generate immutable system logs, vulnerability reports, CAB ticket histories etc. Access Control and SoD correspond to Automated Access Governance that produces approval records and IAM reports. Delay in Workflow is addressed through customized sprints generating digital audit artifacts and compliance dashboards.

The evidence mechanisms confirm that compliance is continuous instead of being retrospective. Automation creates a self-validating delivery model as every pipeline run generates the proof for its own compliance. This answers RQ3 by establishing a closed loop delivery pathway. This finding reinforces the argument that when governance is implemented through automation, it enhances the agility rather than restricting it.

5.3 Relation to Prior Research Works

This qualitative and interpretive study confirms and extends the prior research works that identified automation and traceability as key enablers of regulatory compliance in Agile contexts (Fitzgerald & Stol, 2017; Poth et al., 2019). The earlier studies described the conceptual idea of embedding compliance in software processes. But very few studies provided empirical cross-sector evidence. This research addresses the gap by showing how automation-based compliance is already operational in banking, energy and healthcare sectors.

The results align with the principles of continuous compliance and DevSecOps. It demonstrates their evolution towards RegOps also where the regulatory processes can be operationalized through continuous verification and audit automation. The organizations achieve persistent compliance assurance by integrating governance in code without compromising the release speed. This contributes new empirical validation to the emerging literature on RegOps and the convergence of regulatory and operational objectives in modern software delivery.

5.4 Practical Implications

5.4.1 For DevOps Teams

The teams should embed compliance checks directly into pipelines because every build will produce its own audit evidence in this way. This approach minimises the manual oversight, significantly enhances the transparency and reduces the delivery frictions.

5.4.2 For Audit and Compliance Functions

Auditors can rely on automatically generated records like approval logs, risk dashboards, IAM reports etc., as credible compliance artefacts. This reduces the audit preparation time and shifts the focus from manual verification to continuous assurance interpretation.

5.4.3 For Regulators

Regulators can use these findings and adopt the real-time supervision models using automated evidence as continuous oversight. This approach redefines compliance from being documentation based to evidence-based monitoring.

5.4.4 For Vendors

The tool vendors and platform providers have an opportunity to standardize compliance automation frameworks. Integrating regulatory templates like GDPR, ISO 27001, DORA etc. into pipeline management systems, can enable built-in compliance automation across industries.

5.5 Sector Nuances (Banking, Healthcare and Energy)

The sectoral analysis showed distinct compliance priorities. The banking sector has the highest automation maturity due to DORA and internal audit mandates. The healthcare organizations remain documentation heavy because of strict ISO and patient safety regulations. Still, automated verification is emerging in the healthcare software development industry. The energy sector faces multiple complex regulations which demand the interoperability of governance systems. Despite these differences, all sectors demonstrate a consistent transition towards automation-based regulatory assurance and confirm the universality of C-P-E model.

5.6 Limitations and Alternative Explanations

The findings are based on qualitative data from five interviews and thirteen case studies. All these sources provide the depth of triangulated data from multiple sectors. But the qualitative scope is limited. More semi-structured interviews and a quantitative validation could further confirm the observed relationships. Also, the sectoral and organizational contexts may vary which influences the degree of automation maturity. Moreover, the regulatory landscapes evolve very rapidly, so some identified practices may require periodic adaptation. Despite all these limitations, the triangulated methodology ensures strong validity and replicability across comparable regulated environments.

5.7 Future Works

While this study mainly focused on the regulatory challenges in Agile and DevOps environments, the principles of compliance automation have broader potential applications, particularly in AI and Machine Learning operation (MLOps). As the organizations are increasingly deploying AI-driven systems, new regulatory challenges emerge around algorithmic transparency, data privacy, bias mitigation and explainability.

Future research will therefore extend this framework toward RegOps-MLOps integration which will explore how continuous compliance mechanisms can govern modern development and deployment. In AI contexts, compliance mechanisms must evolve from system-level assurance to model-level governance. Then every model update or retaining cycle can automatically generate traceable evidence of ethical and regulatory conformity. The existing frameworks like GDPR, ISO 42001, the EU Act etc. provide partial guidance, but these are too static for ML pipelines. These do not fully address the dynamic and iterative nature of ML model updates. The future research should explore how continuous verification, evidence generation and compliance-as-code can be extended to govern MLOps workflows.

Therefore, the future works will investigate how regulation can become continuous, adaptive and technically enforced which will mirror the success of DevOps compliance automation. This includes exploring RegOps principles, the key conflicts between regulatory frameworks and AI operations, identifying the constraints, designing AI compliance pipelines and proposing RegOps frameworks that merge regulatory oversight with ML lifecycle management. Such expansion can transform regulatory assurance for AI by making it dynamic, auditable and operationally embedded just like DevOps transformed the software delivery. This direction holds substantial value for both academia and industry in shaping the next generation of compliance automation.

5.8 Chapter Summary

This chapter discussed how automation-driven practices allow regulated organizations to balance the agility and compliance. The findings confirm that the C-P-E framework forms the foundation for governance-by-design, where automation ensures transparency and trustworthiness. Extending this approach into RegOps and MLOps will be critical for addressing the emerging challenges in AI governance which will enable a future of continuous, auditable and ethical compliance automation.

6. Conclusion

The entire research explained in this document explored the ways, in which Agile and DevOps methods can be adopted in the information systems of banking, healthcare and energy sectors. The goal of this research was to figure out which types of hurdles the teams encounter in their workflow, how they handle and fix those hurdles and what evidence they utilize to ensure smooth development and delivery cycle. Five semi-structured interviews were conducted and thirteen case studies were thoroughly studied to get a clear picture of the challenges and adaptations. All data sourced from real organizations who have to regularly deal with regulatory compliance like GDPR, DORA, PII and ISO Standards. Categorizing the data and finding out the most frequent constraints, practices and evidence resulted in proposing a conceptual Constraint-Practice-Evidence (C-P-E) framework which can be useful for the software development and operation teams to blend with compliance in smart ways.

The core constraint categories include limited tools and development environments, strict access control, delayed workflows and regulatory burdens from the organization. There were a number of identified specific constraints under every category which were mostly observed in every case. Those included prohibiting the developers from freely using cloud tools because of data privacy rules, restricting the developers from doing operational deployments, ensuring CAB approval in every stage of the development and deployment cycle and focusing on excessive and paper-based documentation. These are the issues that appear in almost all the sectors, but they mostly affect the industries that are liable for people’s health, money and safety. But the analysis of the data and achieving the main goal of this research showed that these challenges are not unmanageable. The teams can easily adapt their workflows by integrating automation into their work and reducing the effect of these constraints.

The observed adaptive practices showed that making compliance rules part of the daily processes, can result in successful balance of regulations and agility. Automated access governance greatly assists in controlling tools and permissions without any kind of constant manual checks. The automated governance pipelines add bug and risk scanning directly to CI/CD pipelines and reduce the regulatory burdens. Besides, including compliance tasks and adding points for compliance planning in sprints, makes the works faster and the compliance is also ensured. Automated log and reports generation and dashboards can be used as evidence. The approval records and scan reports prove the system security to the auditors. So, the C-P-E framework ties all of these together which shows that compliance can be a built-in mechanism in daily workflow and it is no longer an extra function or burden for the companies.

The resulting outcomes of this research match with other studies about DevSecOps and Continuous Compliance. This research has added some new details from industry expert interviews and case studies across various sectors. One of the most crucial findings was that the banks use the most automation because they have a high frequency of periodic audits and healthcare sector sticks to creating more documentation due to safety reasons. Therefore, the idea of creating governance by design means that compliance is beneficial for organizational security, smooth operation and reliability. This design saves a lot of time in audit preparation and in some cases, it drops the preparation work by half. This proves that agility and regulations can co-exist together. The suggested simple and conceptual C-P-E framework helps the DevOps teams to visualize the way of automating compliance rules without losing any workflow speed. It helps the auditors to assess the trustworthiness of systems with system generated logs and reports over piles of papers. The vendors can also build tools with built-in compliance templates. This study fills the gap in cross-sector examples and shows the patterns that can work in every sector. Further, this research points to future pathways like RegOps for AI systems so that a similar type of automation can handle the bias and ethical rules in MLOps.

Although this research has limitations like fewer interviews and the continuously changing nature of compliance and technology, it provides a big scope for future work and updates. The suggested practices and evidence might need to be updated which can be done through validation of these ideas in the software development teams in the regulated industry. Still, the combination of the interviews and case studies makes the results solid.

In the end, this thesis shows that regulated organizations can embrace Agile and DevOps practices by inserting compliance tasks into coding and deployment stages. Maintaining this balance is the key for modern technology where teams can deliver better software products in a safer way.

References

Zeller, Marc. (2023). DevCertOps: Strategies to Realize Continuous Delivery of Safe Software in Regulated Domain. *In* Proceedings of the 45th International Conference on Software Engineering: Companion Proceedings (ICSE '23 Companion). IEEE, pp. 334–335. [Available at] https://doi.org/10.1109/ICSE-Companion58688.2023.00094

Gupta, Rajeev & Venkatachalapathy, Mekanathan & Jeberla, Ferose. (2019). Challenges in Adopting Continuous Delivery and DevOps in a Globally Distributed Product Team: A Case Study of a Healthcare Organization. *In* Proceedings of the 14th International Conference on Global Software Engineering (ICGSE 2019). IEEE, pp. 30–34. [Available at] https://doi.org/10.1109/ICGSE.2019.00020

Hrgarek, Nadica. (2012). Certification and regulatory challenges in medical device software development. *In* Proceedings of the 4th International Workshop on Software Engineering in Health Care (SEHC 2012). IEEE, pp. 40–46. [Available at] https://doi.org/10.1109/SEHC.2012.6227011

Gallina, Barbara & UL Muram, Faiz & Castellanos Ardila, Julieth. (2018). Compliance of agilized (software) development processes with safety standards: a vision. *In* Proceedings of the 19th International Conference on Agile Software Development: Companion (XP '18 Companion). ACM, pp. 1–6. [Available at] https://doi.org/10.1145/3234152.3234175

Wagner, Torrey & Ford, Thomas. (2020). Metrics to Meet Security & Privacy Requirements with Agile Software Development Methods in a Regulated Environment. *In* Proceedings of the 2020 International Conference on Computing, Networking and Communications (ICNC). IEEE, pp. 17–23. [Available at] https://doi.org/10.1109/ICNC47757.2020.9049681

Lie, Martin & Sánchez-Gordón, Mary & Colomo-Palacios, Ricardo. (2020). DevOps in an ISO 13485 Regulated Environment: A Multivocal Literature Review. *In* Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software

Engineering and Measurement (ESEM 2020). ACM, pp. 1–11. [Available at] https://doi.org/10.48550/arXiv.2007.11295

Laukkarinen, Teemu & Kuusinen, Kati & Mikkonen, Tommi. (2017). DevOps in Regulated Software Development: Case Medical Devices. *In* Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Results Track (ICSE-NIER 2017). IEEE, pp. 15–18. [Available at] https://doi.org/10.1109/ICSE-NIER.2017.20

Ozkan, Necmettin & Kolukısa, Ayça & Filiz, İsmail & Özer, Enis & Gören, Burak. (2020). Harmonizing IT Frameworks and Agile Methods: Challenges and Solutions for the case of COBIT and Scrum. *In* Proceedings of the 2020 Federated Conference on Computer Science and Information Systems (FedCSIS 2020). PTI, pp. 631–638. [Available at] http://dx.doi.org/10.15439/2020F47

Bakalova, Zornitza & Daneva, Maya & Nguyen, Tuan. (2014). Standards compliance helps value creation in agile projects. *In* Proceedings of the 2014 IEEE Eighth International Conference on Research Challenges in Information Science (RCIS 2014). IEEE, pp. 1–6. [Available at] https://doi.org/10.1109/RCIS.2014.6861054

Su, Lixin & Storer, Tim. (2023). A Case Study of DevOps Adoption within a Large Financial Organisation. *In* Proceedings of the 2023 IEEE International Conference on Software Maintenance and Evolution (ICSME 2023). IEEE, pp. 447–457. [Available at] https://doi.org/10.1109/ICSME58846.2023.00053

Leite, Luís & Rodrigues, Daniel & Almeida, Fernando. (2021). The impact of general data protection regulation on software engineering practices. *In* Information and Computer Security 29(1), 41–60. [Available at] https://doi.org/10.1108/ICS-03-2020-0043

Ozcan Top, Ozden & Mccaffery, Fergal. (2019). To what extent the medical device software regulations can be achieved with agile software development methods? *In* The Journal of Supercomputing 75(8), 5223–5260. [Available at] https://doi.org/10.1007/s11227-019-02793-x

Baloun, K. 2013, Implementing agile software development in an ISO 9001:2008 environment. Master’s thesis, California State University, Dominguez Hills.

Mathrani, Anuradha & Wickramasinghe, Shanuka & Jayamaha, Nihal. (2021). An evaluation of documentation requirements for ISO 9001 compliance in scrum projects. *In* The TQM Journal 34(5), 901–921. [Available at] https://doi.org/10.1108/TQM-08-2020-0177

Mukherjee, U. 2024. Paytm's Crisis Management: Regulatory Compliance and Market Impact. *In* Journal of International Economics, vol. 15, no. 1, pp. 82–87.

Moyón, Fabiola & Soares, Rafael & Albuquerque, Maria & Méndez Fernández, Daniel & Beckers, Kristian. (2021). Integration of Security Standards in DevOps Pipelines: An Industry Case Study. *In* Product-Focused Software Process Improvement. PROFES 2021. Lecture Notes in Computer Science, vol. 13126. Springer, Cham, pp. 434–452. [Available at] https://doi.org/10.48550/arXiv.2105.13024

Usman, Muhammad & Felderer, Michael & Unterkalmsteiner, Michael & Klotins, Eriks & Méndez Fernández, Daniel & Alegroth, Emil. (2021). Compliance Requirements in Large-Scale Software Development: An Industrial Case Study. *In* Product-Focused Software Process Improvement. PROFES 2020. Lecture Notes in Computer Science, vol. 12562. Springer, Cham, pp. 209–225. [Available at] https://doi.org/10.1007/978-3-030-64148-1_24

Moyón, Fabiola & Fernández, Daniel & Beckers, Kristian & Klepper, Sebastian. (2021). How to Integrate Security Compliance Requirements with Agile Software Engineering at Scale?. *In* Product-Focused Software Process Improvement. PROFES 2021. Lecture Notes in Computer Science, vol. 13126. Springer, Cham, pp. 69–87. [Available at] https://doi.org/10.48550/arXiv.2105.13404

Jayakody, J.A.V.M.K & Wijayanayake, Janaka. (2024). DevOps Maturity; A Systematic Literature Review. *In* Proceedings of the 2024 International Conference on Software Engineering and Computer Systems (SCSE 2024). IEEE, pp. 1–6. [Available at] https://doi.org/10.1109/SCSE61872.2024.10550493

Tuape, Micheal & Iiyambo, Petrus & Kasurinen, Jussi. (2023). Organizational governance: Resolving insufficient practice and quality expectation in Small Software Companies. *In* Proceedings of the 2022 European Symposium on Software Engineering (ESSE 2022). ACM, pp. 17–24. [Available at] https://doi.org/10.1145/3571697.3571700

Afaneh, S., Mohammad Rasmi Al-Mousa, Hilal Shrif Al-hamid, AL-Awasa, B., Alia, M., Almimi, H. & Alkhatib, A.A. (2023). Security Challenges Review in Agile and DevOps Practices. *In* Proceedings of the 2023 International Conference on Information Technology (ICIT 2023). IEEE, Piscataway, NJ, pp. 1–6. [Available at] https://doi.org/10.1109/ICIT58056.2023.10226018

Su, Lixin & Storer, Tim. (2023). A Case Study of DevOps Adoption within a Large Financial Organisation. *In* Proceedings of the 39th IEEE International Conference on Software Maintenance and Evolution (ICSME 2023). IEEE, pp. 403–413. [Available at] https://doi.org/10.1109/ICSME58846.2023.00053

Rahman, Akond & Helms, Eric & Williams, Laurie & Parnin, Chris. (2015). Synthesizing Continuous Deployment Practices Used in Software Development. *In* Proceedings of the 13th International Conference on Agile Software Development (Agile 2015). IEEE, pp. 1–10. [Available at] https://doi.org/10.1109/Agile.2015.12

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R.C., Mellor, S., Schwaber, K., Sutherland, J., Thomas, D., 2001. Manifesto for agile software development. Agile Alliance. https://agilemanifesto.org/

Bass, L., Weber, I., Zhu, L., 2015. DevOps: A software architect's perspective. Addison-Wesley Professional.

Appendices

Appendix 1. Interview Questions

LUT University

Research Area: Agile-DevOps Challenges and Practices in Regulated Industries

Interview Questions:

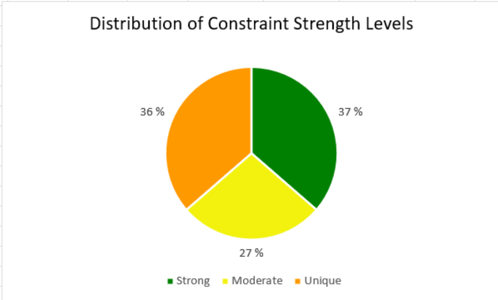
1. What specific regulations significantly impact your team's ability to implement Agile or DevOps practices? Can you describe a specific challenge caused by these regulations?
2. How do regulatory requirements like documentation and verification rules affect your team's ability to maintain continuous integration and deployment?
3. Can you share an example of a regulatory requirement that conflicted with your team's Agile or DevOps goals? How did it impact your work? What was the most significant bottleneck in this case?
4. What processes and workflows have your team modified to balance agility with regulatory compliance?
5. Can you describe a specific strategy or practice that your team has implemented to streamline compliance within Agile-DevOps workflows? What made this strategy effective or challenging?
6. What tools does your team use to support Agile-DevOps practices while ensuring compliance with regulations? How do these tools help to meet regulatory requirements?
7. How do you adapt your sprint planning and release cycles to accommodate regulatory requirements while maintaining Agile principles? What specific changes to sprint planning worked best for compliance?
8. What practical recommendations would you offer to other teams in regulated industries who are trying to adopt Agile-DevOps practices?

Appendix 2. Thematic Coding Summaries

Count of Source ID	Column Labels			
Constraint_Category	Case Study	Interview	Grand Total	Strength
Access Control and SoD	1	5	6	Strong
Coordination conflicts	3	1	4	Moderate
Data privacy and protection		1	1	Unique
Delay in workflow	4	2	6	Strong
External integration control		1	1	Unique
Failed security checking		1	1	Unique
Policy Interpretation Uncertainty	1		1	Unique
Regulatory burden	8	3	11	Strong
Restricted development environment		6	6	Strong
Security and risk compliance	1	4	5	Moderate
Strict Documentation	2	2	4	Moderate
(blank)		19	19	Moderate
Grand Total	**20**	**45**	**65**	

Strength Summary Table

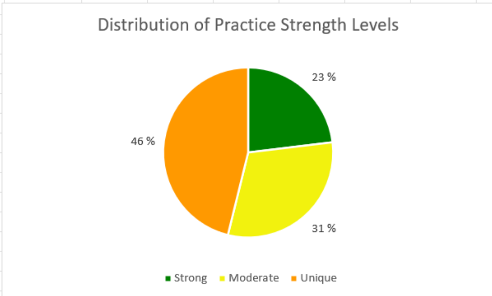
Strength	Count
Strong	4
Moderate	3
Unique	4



Count of Source ID	Column Label			
Practice_Category	Case Study	Interview	Grand Total	Strength
Accepting compliance as part of business		1	1	Unique
Automated access governance		7	7	Strong
Automated governance pipeline	6	9	15	Strong
Cross functional integration	1	1	2	Unique
Customized Workflow	3	5	8	Strong
Human oversight in automation	2	1	3	Moderate
Mapping practice and policy	2		2	Unique
Parallel Documentation	2	2	4	Moderate
Platform Ownership	1	3	4	Moderate
Policy as code automation		1	1	Unique
Policy enforcement	2	2	4	Moderate
Proof of Concept		1	1	Unique
Risk based scaling of process	1		1	Unique
(blank)		12	12	
Grand Total	**20**	**45**	**65**	

Strength Summary Table

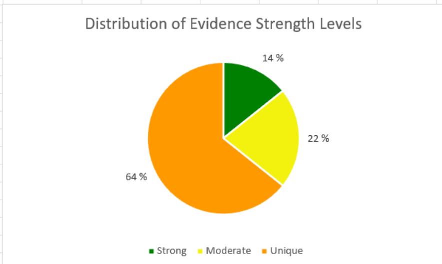
Strength	Count
Strong	3
Moderate	4
Unique	6



Count of Source ID	Column Labels			
Evidence_Category	Case Study	Interview	Grand Total	Strength
Access control evidence		3	3	Moderate
Access control evidence, Immutable system logs		1	1	Unique
Access control policy		1	1	Unique
Approval record and change logs		9	9	Strong
Audit trail documentation	1	2	3	Moderate
Design, testing and incident management documentation		1	1	Unique
Immutable system logs		2	2	Unique
Pipeline logs	1		1	Unique
Scrum artifacts	1		1	Unique
Security and Vulnerability records	2	4	6	Strong
Testing and validation evidence	2		2	Unique
Third party compliance records		3	3	Moderate
Tools and third party control policies		1	1	Unique
Tools and third party control policies, access control evidence		2	2	Unique
(blank)	13	16	29	
Grand Total	**20**	**45**	**65**	

Strength Summary Table

Strength	Count
Strong	2
Moderate	3
Unique	9



Appendix 3. Declaration of AI Usage

LUT University

Declaration of AI Usage

First names: Sumaiya Tabassum

Last name: Antara

Degree Programme: Master's Program in Digital Systems and Service Development

The title of the thesis work in English: Agile-DevOps Integration in Regulated Environments: Challenges and Strategies for Compliance-Driven Industries

Supervisor(s): Professor Kari Smolander (LUT), Junior Researcher Muhammad Hamza (LUT)

I, the undersigned, in line with LUT AI usage recommendations and policy (https://elut.lut.fi/en/completing-studies/rules-and-regulations/ai-based-tools-policies) hereby declare the following regarding the usage of AI tools in the development of the contents of my thesis work:

1a) AI Tool Name: Grammarly AI tool

1b) Purpose of Use: Language maintenance.

1c) Detailed explanation of how and in which sections the tool was used: Throughout the document, grammar and spelling checking have been done using the Grammarly AI tool

2a) AI Tool Name: ChatGPT and Grok AI Tool

2b) Purpose of Use: Understanding the concepts and ideating the structure of research work.

2c) Detailed explanation of how and in which sections the tool was used: Searching for information within constraints, understanding the concepts and testing own knowledge. Ideating the structure of the research work.

I, hereby, declare that no other AI assistance or tools were used in the development of the contents of my thesis work other than the above-mentioned.

Date: 25.11.2025

Signature: Sumaiya Tabassum Antara