Sarcouncil Journal of Engineering and Computer Sciences

ISSN(Online): 2945-3585

Volume- 04| Issue- 10| 2025

Research Article **Received:** 01-09-2025| **Accepted:** 25-09-2025 | **Published:** 10-10-2025

SaaS Verify AI: AI-Enabled Multi-Tenant UAT and Compliance Automation for Regulated SaaS Industries

Kiran Kumar Jaghni

Ness USA Inc, USA

Abstract: In heavily regulated industries, Software-as-a-Service (SaaS) deployments present several important, tenant-specific challenges to the standard User Acceptance Testing (UAT) process, prior to the production release. The UAT process provides confidence that each customer environment will be compliant from both a legal, and operational, perspective, embedded through governance requirements. Traditional UAT methodologies remain predominantly manual, requiring substantial resources and extending release timelines by weeks or months. This article introduces SaaS Verify AI—a novel framework with accompanying reference architecture that leverages intelligent agent technology to transform UAT and compliance validation across distributed multi-tenant SaaS ecosystems. The proposed system autonomously constructs test scenarios, executes verification procedures within isolated tenant environments, and captures tamper-resistant evidence supporting regulatory audits across industry frameworks. This technology supports tenant-controlled release validation and is compliant to regulation, all while providing the potential for reducing release cycles by 60% and operational expenditures on UAT between 30-60% based on similar implementations in the industry. The architecture disrupts the traditional way release validations occur by providing validations as embedded functions instead of external check points, thus creating readiness for audits and increasing confidence in regulated SaaS implementations.

Keywords: SaaS Compliance Automation, AI-Driven Testing, Multi-Tenant UAT, Regulatory Validation, Tenant-Controlled Execution.

INTRODUCTION

Contextual Background

Enterprises in regulated industries using compliance frameworks, including but not limited to U.S. Food and Drug Administration Computer Software Assurance (FDA CSA)(Davidson, J. K. 2023), International Society for Pharmaceutical Engineering Good Automated Manufacturing Practice 5 (ISPE GAMP 5)(International Society for Pharmaceutical Engineering (ISPE), 2022), Health Insurance Portability and Accountability Act (HIPAA)(U.S. Department of Health & Human Services, 1996), U.S. National Archives and Records Administration Policies (NARA policies)(U.S. National Archives and Records Administration, 2023), Drug Supply Chain Security Act (DSCSA)(U.S. Food and Drug Administration, 2025), European Union Falsified Medicines Directive (EU FMD)(European Commission), and other industry regulations, are increasingly transitioning to cloud-based SaaS platforms for enhanced agility, scalability, and reduced infrastructure management(Page, J. 2025). While SaaS adoption has accelerated, customers in compliance-heavy sectors face significant release risk due to frequent updates. Each software upgrade must be validated to ensure compliance, safety, and operational requirements remain intact (Breakers, B. 2012).

SaaS provider Service Level Objectives (SLOs) must guarantee regulatory compliance, functional correctness, and data security and privacy (Vikram, A. 2025; Vanta, 2025; RevTek Capital, 2025; Cohen, Y. 2025; Nicolazzo, S. *et al.*, 2024). UAT remains the primary customer-facing quality gate, but practitioners in regulated sectors often report UAT cycles spanning 4–8 weeks due to increased validation, audit preparedness, and documentation demands (Original Software, 2019). Industry reports reveal that 21% of software testing budgets are spent on intelligent product validation (Capgemini & Sogeti, 2024). These delays impact delivery, time-to-market, and operational costs.

Problem Statement

In compliance-regulated industries, every SaaS platform update must undergo tenant-specific UAT before production deployment. This process validates both functional correctness and regulatory compliance with various frameworks.

Currently, UAT in these sectors is manual, fragmented, and resource-intensive:

- **Cycle Time**: UAT cycles often last 4–8 weeks, delaying releases (Original Software, 2019)
- **Cost Burden**: Testing can consume 20–40% of total project budgets (Global App Testing, 2023), with additional administrative compliance costs of $10,000+ per employee annually (Sharavanan, 2024)
- **Quality Risk**: 75% of organizations repeat UAT cycles due to defects or compliance issues (Original Software, 2019; Capgemini & Sogeti, 2024)

***Corresponding Author:** Kiran Kumar Jaghni

DOI- https://doi.org/10.5281/zenodo.17517021

- **Compliance Mapping Gaps**: Manual mapping of tests to compliance clauses increases audit risk (Italiya, D. 2025)
- **Lack of Tenant Control**: Most automation runs in provider-managed environments, limiting tenant visibility (Camilleri, R. 2024)

Gap Analysis

Existing tools like Tricentis Tosca, Panaya, and mabl.com offer powerful provider-centric automation capabilities (Belcher, D. 2024; Tricentis Tosca, 2025; Panaya, 2025), However, they lack several essential features required in regulated Multi-tenant SaaS environments:

- **Tenant-Specific Automation** – No existing framework dynamically generates tests tailored to each tenant's unique configurations, workflows, and data (Baqar, M., & Khanda, R. 2025)
- **Integrated Compliance Mapping** – Automated mapping of test cases to multiple regulatory frameworks is missing; current mapping is primarily manual (Siena, A)
- **Tenant-Controlled Execution** – Most solutions execute tests in provider-managed environments, creating risks for data sovereignty and tenant visibility (Kumar, R. 2020)
- **Automated Audit Evidence Generation** – No unified mechanism automatically produces immutable, clause-linked compliance artifacts during UAT in a SaaS environment. Even vendors leading in automated evidence collection acknowledge the limitations. For example, StrikeGraph describes AI-powered compliance evidence collection as focused on collecting, organizing, and mapping artifacts across systems, but does not address immutable storage or clause-level linkage of UAT test results in SaaS environments (StrikeGraph. 2023)

No published system integrates all four capabilities into a single, scalable, compliance-aware, AI-enabled UAT model for regulated SaaS deployments (Tricentis Tosca, 2025). To address this gap, this paper proposes SaaS Verify AI as a conceptual framework and reference architecture that unifies these capabilities into a compliance-ready UAT approach.

Table 1: Gap → Conceptual Evaluation Criteria

Gap ID	Problem / Gap (from Background)	Conceptual Evaluation Metric(s)	How could it be assessed in Practice if SaaS Verify AI is implemented
G1	Manual UAT is slow and error-prone	**Cycle-Time Reduction**	Avg. time to complete tenant UAT (baseline vs SaaS Verify AI)
G2	Inconsistent compliance coverage	**Compliance Coverage %**	(# regulatory clauses tested ÷ total applicable clauses) × 100
G3	High cost of manual QA engineers	**Cost Savings %**	(Engineer hours × blended rate) baseline vs SaaS Verify AI
G4	Lack of traceability from tests → compliance clauses	**Traceability Completeness %**	(# evidence artifacts linked to clauses ÷ total tests executed) × 100
G5	No standardized evidence for auditors	**Audit Readiness Time**	Time to compile regulator-ready package (baseline vs SaaS Verify AI)
G6	Vendor-run tests break tenant isolation	**Tenant Control Rate**	% tests executed inside the tenant boundary vs the provider boundary
G7	Poor defect detection in generic scripts	**Defect Detection Recall**	True positives ÷ (True positives + False negatives)
G8	Difficulty scaling across many tenants	**Parallelism / Scalability**	(Max tenants executed in parallel) ÷ baseline capacity
G9	UAT delays block release cadence	**Release Throughput**	# validated releases per quarter baseline vs SaaS Verify AI
G10	Limited ability to reuse tests across frameworks	**Multi-Framework Coverage**	# frameworks supported by mapping engine (HIPAA, FDA CSA, NARA guidelines, DSCSA, and EU FMD, etc)
G11	No versioning of tests/evidence	**Version Traceability**	% test packages & evidence artifacts tagged with semantic version + hash

Publisher: SARC Publisher

Purpose & Scope

This research proposes SaaS Verify AI, an AI-enabled architecture designed to automate UAT and compliance validation for multi-tenant SaaS platforms in highly regulated industries. The proposed conceptual framework addresses inefficiencies, costs, and compliance risks by enabling:

- AI-driven test generation aligned with tenant-specific configurations, business workflows, and regulatory frameworks
- Automated execution of UAT within tenant-controlled runtime environments
- Dynamic compliance mapping, linking test coverage to multiple regulations
- Audit-ready evidence capture supporting regulatory inspections

By bridging the gap between AI-driven testing and compliance-aware SaaS delivery, this framework is designed to achieve benefits reported in prior studies, such as reducing release cycle time by up to 60%, lowering UAT-related operational costs by 30–60%, and strengthening auditor readiness while maintaining trust in production deployments.(QARA Admin, 2024)

Scope: This framework focuses on the pharmaceutical, life sciences, healthcare, food manufacturing, and government services sectors, where compliance validation is mandatory. It covers mapping and validation for major compliance requirements, including FDA CSA, ISPE GAMP 5, HIPAA, NARA guidelines, DSCSA, and EU FMD, etc.

Relevant Statistics

The motivation for SaaS verify AI is reinforced by existing research and industry surveys:

- Multi-tenant SaaS testing requires validating data isolation, tenant-specific configurations, and varied workloads for each tenant—significantly more complex than single-tenant testing (TestGrid, 2025)
- Industry studies show testing consumes up to 40% of development costs, but automation can slash testing expenses by 40–50% (Jurkėnas, R. 2025; QARA Admin, 2024; T. Mostögl, 2025)
- Manual UAT typically spans 2–4 weeks per release, with 75% of organizations repeating UAT cycles due to quality issues (Original Software, 2019)(Quellit, 2025; Moore, N. 2024; Gordon, S. *et al.*, 2022)
- 70% of compliance professionals report shifting from checklist-driven to strategic approaches, with 83% saying compliance adherence is critical for organizational decision-making (Fitzgerald, A. 2024)
- 25% of organizations adding SaaS apps storing sensitive data experienced security or compliance issues, and 12% were penalized for non-compliance (Ohayon, H. 2024)

RESEARCH AND INNOVATIONS

Research Background

The stringent compliance requirements of regulated industries have necessitated specialized, automated solutions, and UAT, which is not exempt from compliance review under an AI solution, continues to be a major bottleneck requiring rigorous testing to comply. Industry surveys confirm that regulated sectors often allocate 10–20% of IT project budgets to UAT activities, with testing cycles lasting several weeks (Original Software, 2019; Capgemini & Sogeti, 2024).

While recent advancements in AI-driven testing have demonstrated the ability of Large Language Models (LLMs) and machine learning to improve test coverage and reduce manual effort (Karhu, K. *et al.*, 2025), commercial tools such as Tricentis Tosca and Panaya remain provider-side, lacking multi-tenant execution, compliance mapping, and audit-ready evidence capture within isolated tenant environments (Tricentis Tosca, 2025; Panaya, 2025; TestGrid, 2025)

Research in multi-tenant SaaS architectures has emphasized tenant isolation and CI/CD integration. Still, current approaches do not extend to tenant-controlled UAT that is dynamically generated, compliance-aware, and securely executed within the tenant's operational boundary.

Novel Contribution

SaaS Verify AI is proposed as a first-of-its-kind conceptual framework and AI-enabled architecture for multi-tenant UAT and compliance automation in regulated SaaS environments. The novel contributions include:

Tenant-Specific AI-Driven Test Generation

AI agents generate dynamic, tenant-specific test cases by interpreting tenant configurations, workflows, and data models. Differentiates from provider-centric tools by localizing the test scope to each tenant's environment.

Automated Multi-Framework Compliance Mapping

Integrated compliance knowledge graph cross-maps test cases to multiple regulatory frameworks

Publisher: SARC Publisher

in real time. Provides traceable compliance evidence with explicit test-to-regulation linkage.

Secure Tenant-Controlled Execution in Isolated Environments
UAT automation runs within each tenant's isolated runtime, ensuring data sovereignty and security
Addresses governance concerns by keeping sensitive data within the tenant boundary

Automated Audit-Ready Evidence Capture
Captures execution logs, screenshots, data validation results, and compliance mappings into immutable, auditor-friendly packages. Supports regulatory inspections with structured, searchable validation evidence

Multi-Tenant Orchestration and Scalability - Runtime Tenant Test Publisher
Enables publishing test scripts to tenant environments, parallel execution, and tenant-isolated execution
Demonstrates a scalable model for compliance-heavy SaaS providers handling hundreds or thousands of tenants

Table 2: Novelty of SaaS Verify AI conceptual framework vs. Existing Solutions

Capability / Feature	Tricentis Tosca	Panaya	Generic AI-Driven Testing Tools	SaaS Testing Architectures (Research)	Compliance Automation Frameworks	SaaS Verify AI (Proposed)
AI-driven test generation	□ Yes – generic functional tests	□ Yes – regression-focused	□ Yes – mostly unit/component scope	□ Not addressed	□ Not addressed	□ Yes – tenant-specific, compliance-aware
Multi-tenant support	□ Single execution context	□ Single execution context	□ Not designed for multi-tenancy	□ Architectural isolation models only	□ Not addressed	□ Yes – designed for multi-tenant SaaS platforms
Tenant-specific test generation	□	□	□	□	□	□ Yes – tests generated per tenant's config, workflows, and data
Compliance mapping to multiple frameworks	□	□	□	□	□ Guidelines exist, but no automation	□ Automated cross-mapping to FDA CSA, ISPE GAMP 5, HIPAA, NARA, DSCSA, EU FMD, etc.
Tenant-controlled execution	□ Provider-controlled	□ Provider-controlled	□ Central execution only	□ Isolation concepts, but no UAT automation	□ Not addressed	□ Execution within each tenant's isolated environment
Automated audit-ready evidence capture	□ Manual or limited	□ Manual or limited	□ Not supported	□ Not supported	□ Not automated	□ Fully automated, structured for audits
Parallel execution across tenants	□	□	□	□	□	□ Orchestrated, scalable multi-tenant execution
Integration with CI/CD	□ Yes	□ Yes	□ Yes	□ Yes	□ Not specified	□ Yes – with compliance validation gate
Target domain	Broad industries	ERP/CRM change validation	Generic software testing	SaaS deployment & scaling	Compliance guidance only	Regulated SaaS industries and Non-Regulated for UAT automation.

Publisher: SARC Publisher

Operational benefits	Faster regression testing	Change impact reduction	Test generation speed	Scalability in SaaS ops	Compliance assurance	Potential improvements based on the literature. Target to 60% cycle time reduction, Target to 30–60% cost savings, and audit readiness based on the prior work literature

METHODOLOGY

- **Research Approach**
- This study adopts a design science research methodology (DSRM) (Peffers, K. *et al.*, 2017), combining problem identification, solution design, and iterative refinement. The approach includes
- **Problem Identification and Motivation** – Analysis of regulated SaaS UAT practices
- **Defining Objectives** – Establishing functional and non-functional requirements
- **Design and Development –** Formulating the SaaS Verify AI conceptual framework and architecture
- **Demonstration –** Conceptual demonstration through reference models and illustrative workflows
- **Assessment – Definition of evaluation metrics** (Cycle Time Reduction, Cost Reduction, Compliance Coverage, Execution Accuracy, and Audit Readiness Time)
- **Communication –** Scholarly communication of the conceptual design, including the definition of evaluation metrics and a proposed reporting approach for future validation.

Data Flow & Process Steps

The proposed conceptual dataflow consists of:

- **Input Gathering** – Tenant metadata, configurations, and compliance frameworks
- **AI-Based Test Generation** – LLM-based engine to create test cases aligned to tenant workflows and mapped compliance clauses.
- **Test Deployment** – Orchestrator publishes test scripts to tenant environments.
- **Execution & Monitoring** – Envisioned parallel execution, with real-time tracking
- **Evidence Capture & Packaging** – Outputs storage in an immutable format, indexed by tenant, release version, and compliance rules
- **Result Consolidation** – Aggregated compliance and validation reports.
- For details, see Figure -1: Conceptual Data Flow diagram

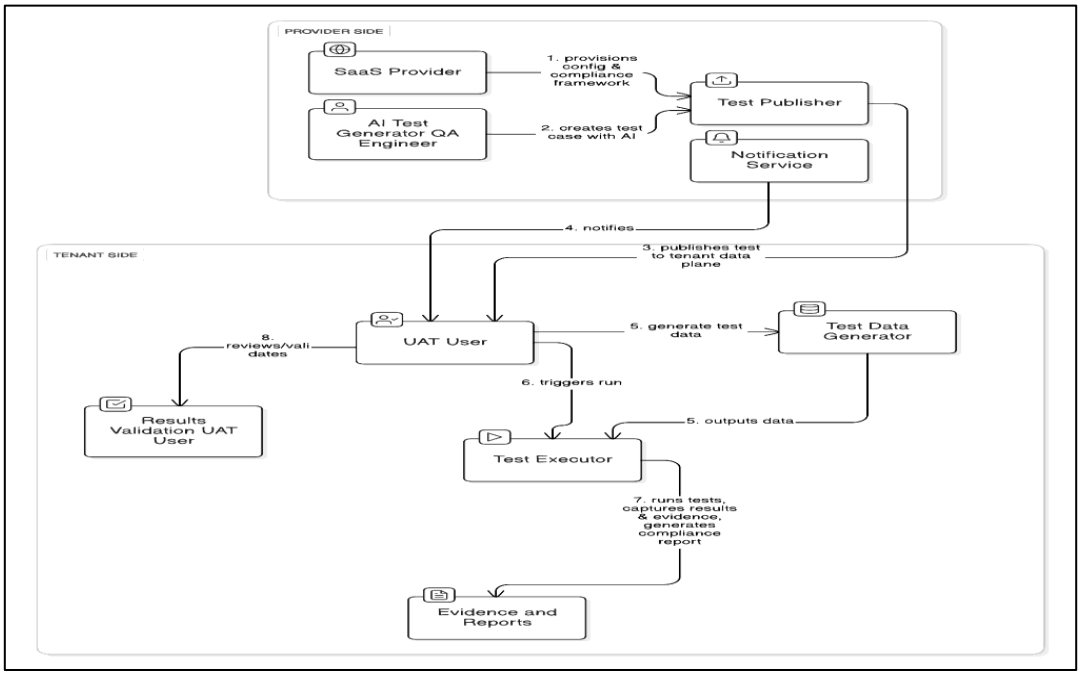

Figure -1: Conceptual Data Flow diagram

Publisher: SARC Publisher

Architectural Design
Below are the building blocks of the proposed conceptual software architecture framework.

Compliance Framework & Rules and Tenant Configuration
Framework to configure compliance rules per tenant

Control Plane User Interface
The QA Engineer is to generate the use case-specific tests.

AI-Driven Test Generator
LLM-based agents analyze tenant configurations to generate functional and compliance test cases.
Supports multiple test types, including functional validation and security compliance tests

Tenant Test Publisher
Publishes the generated tests as per the configured compliance rules per tenant at runtime.

Test Repository
Stores test scripts, Test data, results, evidence references, and metadata for dynamic test generation.

Compliance Mapping Engine
Maintains a compliance knowledge graph linking test cases to specific regulatory clauses
Performs real-time validation of coverage gaps

Audit Evidence Management
Captures execution artifacts into immutable, auditor-ready packages
Stores artifacts in compliance with regulatory retention policies

Tenant-Controlled Execution Layer
Deploys test packages within each tenant's isolated runtime environment
Ensures data sovereignty and eliminates cross-tenant interference

Tenant-Test Web Console
Interface for UAT users to author tests and view results/evidence.
For details, see Figure 2: Proposed Conceptual Framework.

The Detailed high-level proposed conceptual software architecture consists of the following core components.

Control Plane Components:
A library of all compliance rules across different compliance frameworks.

- The tenant configurations are required to make the Tenant provisioning and onboarding successful and the tenant operational.
- Software quality assurance engineers use the Control Plane UI to generate and manage the test suites.
- The AI Test Generators are responsible for generating the test cases as per the requirements specs and supplied compliance rules.
- The tenant test publisher is responsible for publishing the Test cases/scripts to the respective tenant.
- The Notification service listens to the events and notifies the user as per the notification subscriptions. E.g., When news tests are created, the UAT User will receive a notification, who is responsible for sending messages.

Tenant Data Control Plane Components:
The Testcase/Test scripts published from Tenant Test Publisher are stored in the Tenant Test Repository.

The UAT user can access the Test via the Tenant Web Console, and the user can trigger the Test execution, and the Test Executor starts the execution of the tests.

The Rule engine works in conjunction with Test Executor, Compliance Engine, AI Test Generator, and Test Repository. The Rule Engine can route requests from the Test Executor to the Compliance Engine for analysis of the test cases /scripts based on the Compliance test coverage score.

The compliance engine analyzes the test scenarios and verifies the compliance mappings. It also identifies the compliance gaps in the mappings and auto-corrects the mappings. The compliance engine also analyzes missing compliance tests for a given Test scenario and further requests the rule engine to take the subsequent actions.

The Rule engine, based on the compliance engine request, will request the Control planes AI test Generator to generate additional tests and wait for the test generation to be completed. Once the new tests are received, they will be added to the existing test runs.

The Rule engine saves the newly generated compliance tests (based on the compliance engine gap analysis outcome) to the Test Repository.

Now the Test Executor requests Test Data Generator to generate the Test Data for the given

Publisher: SARC Publisher

Test cases/ Test scripts and executes the tests using the test data generated from the Test Data Generator.

The test results of the tests for a given test run will be stored in the Test repository.

The Test Executor requests the compliance audit engine to generate a compliance audit report and save it against the test run in the Evidence storage.

The other test evidence is processed by the Test Evidence collector and stored in the Evidence storage against the test run.

The Test evidence is available for the UAT user and accessed via the Tenant Web Console for verification. Also, all other test results are accessed from Tenant Web Console.

The Compliance Audit reports are available for the UAT user and accessed via Tenant Web Console for verification.

For details, see Figure 3: Proposed Conceptual Architecture.

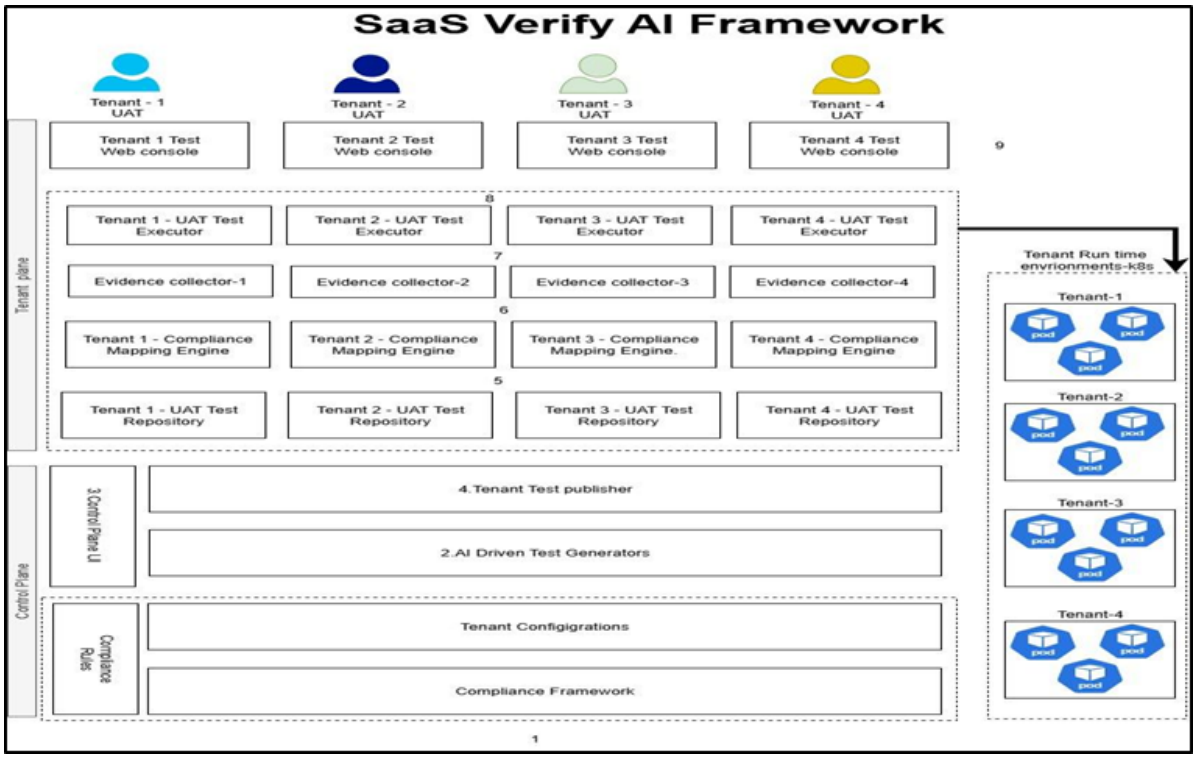

Figure 2: Proposed Conceptual Framework

Publisher: SARC Publisher

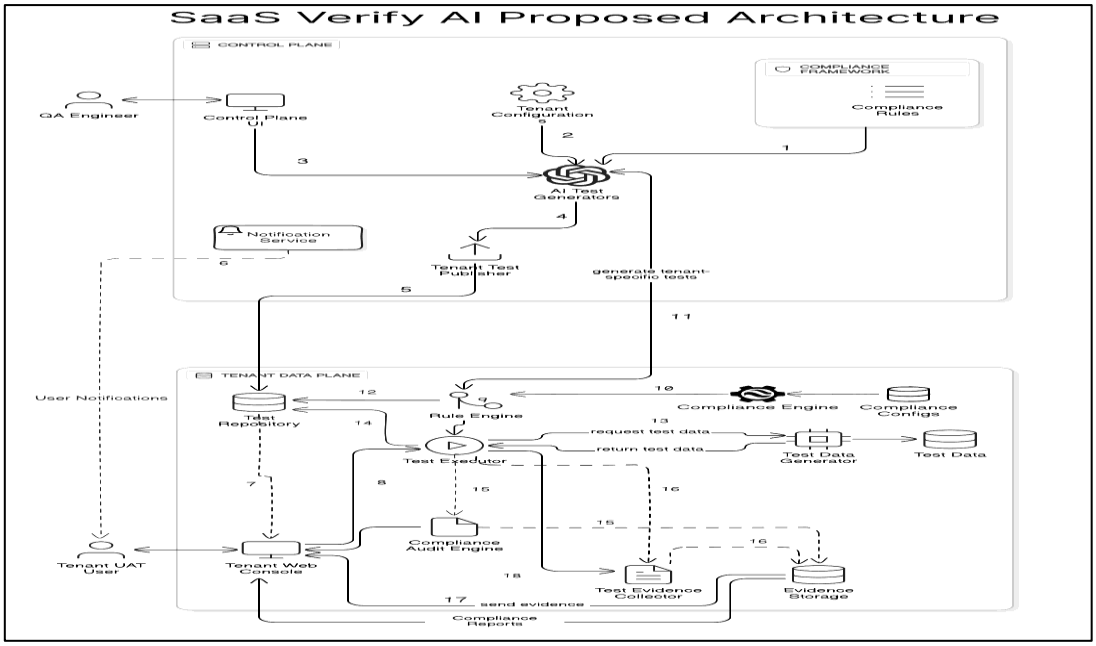

Figure 3: Proposed Conceptual Architecture

Evaluation Plan

The evaluation plan defines a conceptual framework for assessing technical feasibility, compliance coverage, and operational efficiency of SaaS Verify AI. These evaluation criteria are proposed as design objectives rather than reported experimental results.

Testbed: A simulated Multi-tenant SaaS environment with representative workloads from government agencies, food/pharmaceuticals, and healthcare domains is envisioned as a reference model for evaluation.

Metrics (Design Objectives):

- Cycle Time Reduction (% decrease in UAT execution time)
- Cost Savings (% reduction in resource hours vs. manual UAT)
- Compliance Coverage (% of regulatory clauses mapped and tested)
- Execution Accuracy (% of tests producing expected results)
- Audit Readiness Time (time to produce a complete evidence package)

Baseline Comparison: Evaluation would compare the projected performance of SaaS Verify AI against traditional manual UAT and provider-controlled AI test tools.

Execution Paths:

- Baseline path: Manual with manual execution.
- Proposed Path: SaaS verify AI conceptual flow for test generation, test publication, compliance mapping, and test execution.

Metrics Hub & Analytical Services: Conceptually designed to capture and report on these metrics, providing insight into potential gains if implemented.

Table 3: Evaluation Metric Crosswalk

Gap ID	Problem / Gap (from Background)	Evaluation Metric(s)	Definitions of how performance would be measured in practice is SaaS verify AI is implemented
G1	Manual UAT is slow and error-prone	**Cycle-Time Reduction**	Avg. time to complete tenant UAT (baseline vs SaaS Verify AI)
G2	Inconsistent compliance coverage	**Compliance Coverage %**	(# regulatory clauses tested ÷ total applicable clauses) × 100
G3	High cost of manual QA engineers	**Cost Savings %**	(Engineer hours × blended rate) baseline vs SaaS Verify AI
G4	Lack of traceability from tests → compliance clauses	**Traceability Completeness %**	(# evidence artifacts linked to clauses ÷ total tests executed) × 100
G5	No standardized evidence for auditors	**Audit Readiness Time**	Time to compile regulator-ready package (baseline vs SaaS Verify AI)

Publisher: SARC Publisher

G6	Vendor-run tests break tenant isolation	**Tenant Control Rate**	% tests executed inside the tenant boundary vs the provider boundary
G7	Poor defect detection in generic scripts	**Defect Detection Recall**	True positives ÷ (True positives + False negatives)
G8	Difficulty scaling across many tenants	**Parallelism / Scalability**	(Max tenants executed in parallel) ÷ baseline capacity
G9	UAT delays block release cadence	**Release Throughput**	# validated releases per quarter baseline vs SaaS Verify AI
G10	Limited ability to reuse tests across frameworks	**Multi-Framework Coverage**	# frameworks supported by the mapping engine (HIPAA, FDA CSA, …)
G11	No versioning of tests/evidence	**Version Traceability**	% test packages & evidence artifacts tagged with semantic version + hash

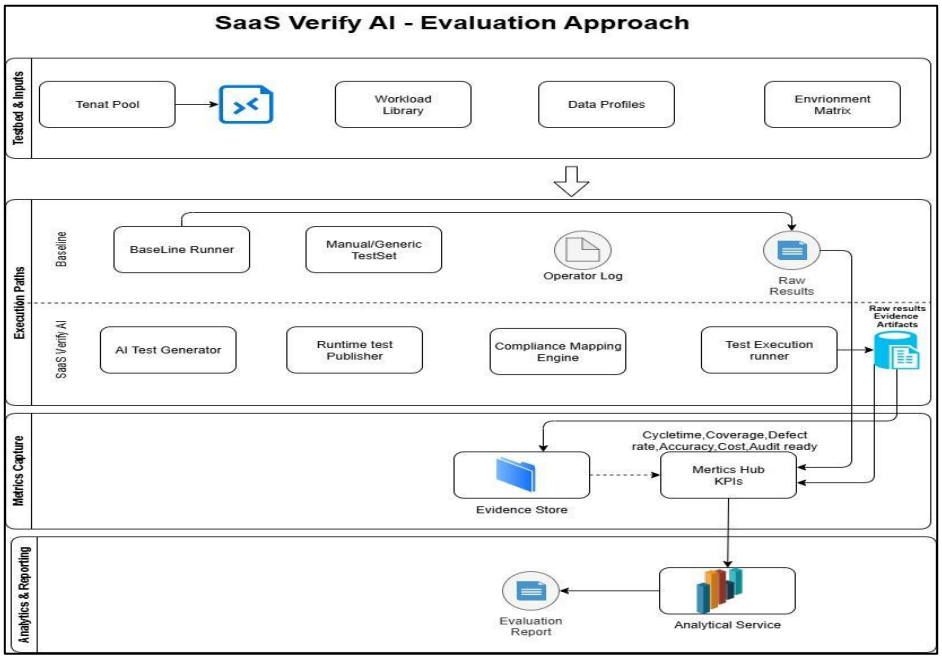

Figure 4: Evaluation Approach

Table 4: Metric Definitions & Formulas

Metric	Operational Definition	Formula / Measurement Method	SaaS Verify AI is expected if implemented
Cycle-Time Reduction	Time to complete UAT per tenant release	(Avg. UAT duration_baseline – Avg. UAT duration_SaaS) ÷ Avg. UAT duration_baseline × 100	40–60% faster
Compliance Coverage %	Extent of regulatory clauses validated	(# clauses tested ÷ total applicable clauses) × 100	≥95% (vs 50–70% baseline)
Cost Savings %	QA engineer + infra cost saved	(Cost_baseline – Cost_SaaS) ÷ Cost_baseline × 100	30–50% lower
Execution Accuracy	Correctness of test verdicts vs ground truth	(True Positives + True Negatives) ÷ Total cases	≥98% (vs 85–90% baseline)
Audit Readiness Time	Time to generate regulator-ready evidence/report	End-to-end elapsed minutes from run completion to packaged evidence	<1 day (vs 1–2 weeks baseline)
Defect Detection Recall	Fraction of seeded defects found	TP ÷ (TP + FN)	≥90% (vs 60–70% baseline)
Parallelism / Scalability	Ability to handle multiple tenants simultaneously	Max concurrent tenants validated	Linear scaling; 10× baseline

Publisher: SARC Publisher

Multi-Framework Coverage	Regulatory frameworks mapped	Count of supported frameworks in the knowledge graph	4–6 frameworks supported (vs 1–2 baseline)
Version Traceability	Evidence + tests tied to semantic versions	(# artifacts with version+hash ÷ total artifacts) × 100	100%

Validation Approach

Validation of SaaS Verify AI is currently conceptual. The framework outlines potential pathways for future empirical validation:

- **Internal Validation(Proposed)** – Controlled environment testing using synthetic datasets to explore feasibility
- **External Validation(Proposed)** – Pilot deployment scenarios with industry partners in regulated SaaS environments.
- **Expert Review(Conceptual)** – Structured feedback from compliance officers, QA leads, and SaaS architects.
- **Iterative Refinement(Future work)** – Adjustments to the framework based on outcomes of the future validation efforts.

COMPARATIVE INSIGHT

What Materially Changes vs. Today

- From people-authored test packs → AI-generated, tenant-specific suites(conceptual design)
- From generic regression → compliance-mapped validation
- From provider-controlled runners → tenant-controlled execution
- From ad-hoc screenshots/logs → immutable, packaged audit artifacts

Best Fit Scenarios

Choose Manual/Traditional when:

- One-off, low-risk releases or very small tenants with minimal compliance scope
- Highly novel features requiring exploratory, domain-expert testing

Choose Provider-Side AI Tools when:

- Fast provider-run regression on largely uniform configurations is needed
- Compliance linkage is handled outside tooling

Choose SaaS Verify AI when:

- Tenants have meaningful configuration variance and regulated workflows
- Tenants have non-regulated workflow variance.
- Tenant controlled execution is required (e.g.PHI/records constraints)
- Systematic, clause-mapped evidence at scale across many tenants is needed

Compromises and Methods to Mitigate Risks

- **Model Drift / False Positives:** These risks can be mitigated through the addition of checkpoints, and by including a human in the loop in oversight activity.
- **Compliance graph maintenance**: Conceptually addressed via versioned knowledge graph and diff-based updates
- **Orchestration complexity**: May be managed via quota-based schedulers and per-tenant resource limits
- **Change management**: Supported in principle by transparent rationale per test and preview diffs

Refer to Table 4: Comparative Insights for additional details.

Head-to-head on critical dimensions

Table 5: Comparative Insights

Dimension	Manual / Traditional UAT	Provider-Side AI Tools	SaaS Verify AI (Proposed)
Test authoring	Human, slow, uneven	AI-assisted, generic	**AI-assisted, tenant-specific**
Compliance linkage	Spreadsheet/manual trace	Limited/indirect	**Automated, multi-framework mapping**
Execution locus	Tenant or shared lab	Provider infra	**Tenant-isolated runtime (per-tenant)**
Evidence	Manual snapshots	Partial logs	**Conceptually designed to generate automated, audit-ready packages**
Multi-tenant scale	Serialized, bottlenecked	Not tenant-aware	**Parallelized orchestration across tenants**
Data	High handling risk	Data egress to the	**Data stays in the tenant boundary**

Publisher: SARC Publisher

governance		provider	
Change velocity	4–8 week UAT windows	Faster, but generic	**Conceptually designed to reduce UAT cycle times, with prior studies suggesting potential reductions of 40–60%.**
Total cost of UAT	High human hours	Tool + services	**Lower human hours; platform amortized**

POTENTIAL APPLICATIONS

The proposed conceptual framework and architecture of SaaS Verify AI could be applied across multiple regulated sectors, contingent on future implementation and pilot validation. Illustrative application domains include:

- **Pharmaceutical & Life Sciences** – Automating GxP and FDA CSA validation
- **Healthcare** – Ensuring HIPAA-compliant EHR updates
- **Food Manufacturing** – Validating DSCSA/EU FMD compliance for supply chain traceability
- **Government Land Records** – Running clause-mapped tests to meet statutory indexing rules
- **Multi-Jurisdiction Adaptation** – Re-running compliance test suites for tenants under different regulatory regimes
- **Disaster Recovery Validation** – Testing recovery workflows to ensure business continuity
- **Third-Party System Validation** – Assessing vendor integrations before connecting to production
- **Tenant Onboarding** – Rapid compliance validation for new customers
- **Pre-Audit Mock Runs** – On-demand execution of targeted compliance tests

BROADER IMPLICATIONS

Environmental Impact

Conceptually, SaaS Verify AI may reduce the environmental footprint by minimizing travel for audits, cutting compute waste through optimized test execution, and eliminating paper-based audit records.

Economic Impact

If implemented, SaaS Verify AI is expected to yield cost savings by 30–60%, shorten release cycles by up to 60%, and eliminate multi-week audit preparation efforts. Regulatory compliance consumes 1.3–3.3% of total wage costs for U.S. firms, averaging $10,000 per employee annually (National Bureau Of Economic Research, 2023).

Social Effects

SaaS Verify AI may strengthen public trust in digital services, improve service reliability, standardize quality across tenants, enable transparency through immutable audit evidence, and support workforce upskilling.

Long-term Outlook

If adopted, the proposed conceptual framework could help regulated SaaS environments align with several emerging industry trends:

- **Continuous Compliance as Default** – Moving toward real-time compliance verification
- **AI-Governed Digital Services** – Increasing demand for explainable, compliance-aware AI
- **Multi-Jurisdictional Complexity** – Adapting validation to diverse regional regulations
- **Zero-Touch Releases** – Evolving toward fully automated validation with zero manual intervention
- **Integration with ESG Goals** – Linking compliance automation to sustainability reporting
- **Cyber-Compliance Convergence** – Unifying security validation with regulatory compliance checks

CONCLUSION

This paper has presented SaaS Verify AI, an innovative framework with corresponding reference architecture designed to address the persistent challenges of compliance-regulated UAT within multi-tenant SaaS environments. The architecture integrates four essential capabilities: AI-powered tenant-specific test generation, comprehensive multi-framework compliance mapping, secure tenant-controlled execution mechanisms, and automated audit evidence capture.

For SaaS providers, the runtime tenant test publishing approach represents a transformative path toward compliance-aware, dynamically generated testing executed within isolated tenant environments—effectively recasting compliance as an embedded, continuous service rather than a release impediment.

Publisher: SARC Publisher

To a larger degree, this development also foreshadows a future where regulated SaaS platforms are increasingly differentiated based on design principles that provide a compliance-safe-to-design context and supports faster updates with more transparency and safety. While framed as an abstract concept, this research lays the groundwork for future empirical research study, testing and iterations with the end goal of making compliance a competitive advantage versus simply a regulatory burden.

REFERENCES

1. Davidson, J. K. "Advancing the transition to computer software assurance: Responding to the FDA draft guidance for production and quality system software." *FDLI Update* (2023): 22.
2. International Society for Pharmaceutical Engineering (ISPE), GAMP 5: *A Risk-Based Approach to Compliant GxP Computerized Systems, 2nd ed.,* (2022).
3. U.S. "Department of Health & Human Services, Health Insurance Portability and Accountability Act of 1996 (HIPAA). Covered Entities and Business Associates. (1996)
4. PCI, "Security Standards Council, PCI DSS v4.0 Requirements and Security Assessment Procedures, 2022." *PCI Security Standards Overview.*(2022)
5. U.S. National Archives and Records Administration, *General Records Schedules and Records Management Policy*, (2023).
6. U.S. Food and Drug Administration, Drug Supply Chain Security Act (DSCSA),—*Title II of the Drug Quality and Security Act*, (2025).
7. European Commission, "Regulation (EU) 2019/1020 on the enforcement of compliance with Union harmonization legislation, relating to the Falsified Medicines Directive (FMD)."
8. Ponemon Institute, *The True Cost of Compliance with Data Protection Regulations*, (2011).
9. Hexaware Technologies, "Test automation solution to cut test cycle time by 70% for a global professional services firm," *Hexaware*.
10. Sanghavi, C. *et al.,* "How Drata's continuous compliance solution helps SaaS providers streamline compliance on AWS," AWS Partner Network Blog, Nov. (2023).
11. Page, J. "46 SaaS Industry Stats and Insights for 2024 and Beyond," *SaaS Academy*, (2025).
12. Breakers, B. "Why Automated Software Testing for SaaS Products is a Must in 2025," *Beta Breakers Blog*, (2025).
13. Vikram, A. "Securing the Cloud: Best Practices for Cloud and Infrastructure Services in Regulated Industries," *Practical Logix*, (2025).
14. Vanta, "Your Guide to SaaS Compliance: Key Areas and Best Practices," *Cloud Security Alliance Blog*, (2025).
15. RevTek Capital, "Regulatory Compliance and Privacy in SaaS," (2025).
16. Cohen, Y. "PII Compliance Checklist: Best Practices for SaaS Security," *Sentra.io*, (2025).
17. Nicolazzo, S. *et al.,* "Service Level Agreement and Security SLA in Cloud Computing," (2024).
18. Original Software, "User Acceptance Testing Survey Report," *Original Software*, Jul. (2019).
19. Capgemini & Sogeti, "World Quality Report 2024-25," *World Quality Report*, Oct. (2024).
20. Italiya, D. "What Is UAT in Software Development? Beginner Guide," *TST Technology Matrix*, (2025).
21. Sharavanan, "Key Compliance Statistics & Insights for 2025," *Zluri Blog*, (2024).
22. Global App Testing, "32 Software Testing Statistics for Your Presentation in 2025," *Global App Testing Blog*, (2023).
23. Original Software, "User Acceptance Testing Survey Report," *Original Software*, Jul. (2019).
24. Kosenkov, O., Elahidoost, P., Gorschek, T., Fischbach, J., Mendez, D., Unterkalmsteiner, M., ... & Mohanani, R. "Systematic mapping study on requirements engineering for regulatory compliance of software systems." *Information and Software Technology* 178 (2025): 107622.
25. Camilleri, R. "Data security in cloud-centric multi-tenant databases." (2024).
26. Baqar, M., & Khanda, R. "The Future of Software Testing: AI–Powered Test Case Generation and Validation." *Intelligent Computing-Proceedings of the Computing Conference*. Cham: Springer Nature Switzerland, (2025).
27. Siena, A. "Mapping Software Testing Practices to Compliance Requirements," conference Paper in Computer Science, University of Toronto.
28. Kumar, R. "Multi-Tenant SaaS Architectures: Design Principles and Security

Publisher: SARC Publisher

Considerations." *Journal of Architecture and Civil Engineering* 5.2 (2020): 49-62.
29. StrikeGraph. “Automated to AI-powered evidence collection in compliance: Benefits, challenges, & trends.” *StrikeGraph.* (2023, December 12).
30. Belcher, D. “Recognizing 7 Years of AI Innovation in Test Automation,” *Mabl Blog,* Mar. 14, (2024).
31. “Tricentis Tosca,” (2025).
32. “Panaya,” (2025).
33. TestGrid, “Comprehensive Guide: SaaS Software Testing,” *TestGrid Blog,* Apr. (2025).
34. Amazon Web Services, “REL 3: How are you testing the multi-tenant capabilities of your SaaS application?” *AWS SaaS Lens,* (2025).
35. Kataria, A. “7 Challenges in Multi-Tenancy Testing and Their Solutions,” *Net Solutions Blog,* Apr. 10, (2020).
36. Pinto, V. H. S. C., Oliveira, R. R., Vilela, R. F., & Souza, S. R. "Evaluating the User Acceptance Testing for Multi-tenant Cloud Applications." CLOSER. (2018).
37. Jurkėnas, R. “Understanding the Cost of Software Testing in Software Development: A Complete Breakdown,” *IdeaLink Blog,* May 13, (2025).
38. QARA Admin, “The Benefits of Automation Testing: A Game Changer for Modern Enterprises,” *QARA Enterprise Blog,* Sep. 13, (2024).
39. T. Mostögl, “Automated Testing: Implementation Strategy and ROI Analysis for Enterprises,” *Virtuoso QA Blog,* (2025).
40. Global App Testing, “32 Software Testing Statistics for Your Presentation in 2025,” *Global App Testing Blog,* (2023).
41. Quellit, “Understanding the ROI of User Acceptance Testing Automation,” *Quellit Blog,* (2025).
42. Moore, N. “User Acceptance Testing (UAT): How to Navigate the Last Development Hurdle Before Production,” *Qase Blog,* (2024).
43. Gordon, S., Crager, J., Howry, C., Barsdorf, A. I., Cohen, J., Crescioni, M., ... & Electronic Patient-Reported (ePRO) Consortium, PRO Consortium. "Best practice recommendations: user acceptance testing for systems designed to collect clinical outcome assessment data electronically." *Therapeutic Innovation & Regulatory Science* 56.3 (2022): 442-453.
44. Fitzgerald, A. “110 Compliance Statistics to Know for 2025,” *Secureframe Blog,* (2024).
45. Ohayon, H. “The Essential Guide to SaaS Compliance,” *Suridata Blog,* (2024).
46. Jennings, D. “The Rise of SaaS and the Alarming Gap in Data Protection,” *HYCU Blog,* (2024).
47. U.S. Department of the Treasury, *The Financial Services Sector’s Adoption of Cloud Services,* (2023).
48. Clark, B. *et al.,* “Weighing opportunity and risk for cloud adoption in healthcare,” *Becker’s Hospital Review,* (2017).
49. GHX, “Nearly seventy percent of hospitals and health systems to adopt cloud-based supply chain management by 2026,” *Press release,* Sept. 26, (2023).
50. U.S. General Services Administration, “GSA celebrates major milestones in FedRAMP cloud authorizations,” News Release, Aug. 11, (2025).
51. Adler, B. “The latest cloud computing trends: Flexera 2025 State of the Cloud Report,” *Blog,* Mar. 19, (2025).
52. Karhu, K., Kasurinen, J., & Smolander, K. "Expectations vs Reality--A Secondary Study on AI Adoption in Software Testing." *arXiv preprint arXiv:2504.04921* (2025).
53. Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. "A design science research methodology for information systems research." *Journal of management information systems* 24.3 (2007): 45-77.
54. National Bureau Of Economic Research, “Tracking the Cost of Complying with Government Regulation,” *NBER Digest,* (2023).
55. Stevenson, R. “115 Compliance Statistics You Need To Know in 2025,” *Drata Blog,* (2025).

Source of support: Nil; **Conflict of interest:** Nil.

Cite this article as:
Jaghni, K. K. " SaaS Verify AI: AI-Enabled Multi-Tenant UAT and Compliance Automation for Regulated SaaS Industries." *Sarcouncil Journal of Engineering and Computer Sciences* 4.10 (2025): pp 45-57.

Publisher: SARC Publisher