ISSN - 2278-9723

Research Article

Execution Trace Forensics in Low-Code Platforms for Compliance-Critical Workflows

VISHNU VARDHAN REDDY KAVULURI[1], SRINIVASARAO BANDLA[2], MAHESWARA RAO GORUMUTCHU[3], JASWANTH KUMAR MANDAPATTI[4], NARESHKUMAR JAGADHABI[5]

[1]Ness USA INC, United States, Email: vishnu.kavuluri@gmail.com
[2]Deloitte Consulting LLP, United States, Email: Bandla.srinivas10@gmail.com
[3]HYR Global Source Inc, United States, Email: gmrmails@gmail.com
[4]Advent Health, United States, Email: jash.209@gmail.com
[5]Compnova Inc, United States, Email: nrkumar544@gmail.com

Received: 16.01.22, Revised: 05.02.22, Accepted:20.03.22

ABSTRACT

Execution trace forensics in AI-assisted low-code platforms has become critical for ensuring compliance, transparency, and reliability in automated workflow environments. While low-code systems enable rapid software assembly, they introduce complex, multi-layered execution behaviors that obscure traceability and complicate forensic analysis. Existing approaches rely on conventional logging mechanisms, which are insufficient for identifying structural, semantic, temporal, and control-flow anomalies across distributed workflows. This study addresses this gap by proposing a formal forensic framework that models execution traces as state-transition sequences and quantifies deviations using anomaly-specific metrics across workflow stages. The results reveal that execution trace deviations exhibit stage-dependent and multi-dimensional characteristics, with transformation, orchestration, and integration layers acting as primary sources of instability. A heatmap-based analysis further demonstrates the differential distribution of anomaly types across workflow stages, enabling precise identification of critical deviation points. The proposed framework enhances forensic reconstruction, supports compliance validation, and provides a foundation for developing adaptive and real-time anomaly detection systems in low-code environments.

Keywords: Execution trace forensics, low-code platforms, anomaly detection, compliance systems

1. Introduction

Execution trace forensics in low-code platforms is becoming a critical requirement for compliance-critical workflows, where auditability, non-repudiation, and evidentiary reconstruction must be preserved under dynamic system conditions. In regulated environments, execution traces act as primary evidence for validating system behavior and decision pathways. Similar traceability requirements are observed in telemedicine systems, where interaction logs must ensure accountability and compliance under distributed execution conditions [1]. These parallels emphasize the need for structured forensic mechanisms capable of interpreting execution traces generated by automated platforms.

AI-assisted software assembly introduces layered execution complexity due to automated orchestration and dynamic workflow composition. Low-code platforms generate multi-stage pipelines involving transformations, validations, and decision routing. Comparable behavior is observed in deep learning–based diagnostic pipelines, where multiple processing stages interact and obscure traceability if not explicitly modeled [2]. Additionally, system-level variability studies show that inconsistencies in processing pipelines can influence downstream outputs even when upstream logic appears correct [3]. This makes forensic reconstruction essential for identifying hidden deviations.

The propagation of anomalies across execution stages further complicates forensic analysis. Studies on behavioral and environmental variability demonstrate that system outputs can change significantly under varying operational conditions, even when underlying logic remains constant [4]. In low-code platforms, frequent updates to workflows, user-defined rules, and configuration parameters introduce variability that affects execution traces. Structural inconsistencies in data representation further amplify this effect, as observed in microbiological systems where encoding variations influence analytical outcomes [5].

Feature transformation and intermediate representations play a crucial role in determining execution trace fidelity. Research in biomedical and pharmacological systems highlights that variations in feature encoding and transformation pipelines can significantly alter system outputs [6]. In low-code workflows, transformations such as mapping rules, expression evaluations, and integration adapters define how data flows through execution stages. Any inconsistency in these transformations can distort trace semantics and reduce the reliability of forensic analysis.

The integration of distributed services and external systems introduces additional complexity in trace correlation. Studies on heterogeneous data environments show that fragmented execution pathways across multiple systems complicate reproducibility and validation [7]. In enterprise low-code platforms, workflows often span APIs, microservices, and legacy systems, leading to fragmented traces that require correlation across multiple layers. Legacy system constraints and historical data formats further contribute to inconsistencies in trace interpretation [8].

Execution trace interpretation is also influenced by evolving system architectures and data conditions. Research in diagnostic imaging systems demonstrates the importance of traceability in validating multi-stage computational processes [9]. Similarly, public health studies indicate that evolving data collection frameworks can introduce variability in analytical outcomes [10]. In low-code environments, evolving schemas and changing data flows introduce additional challenges in maintaining consistent execution traces.

Further complexity arises from classification and decision-making pipelines embedded within automated systems. Studies on disease detection and classification models show that system outputs depend heavily on stable input representations and consistent processing logic [11]. Inconsistent data handling or transformation errors can lead to incorrect outputs while preserving syntactic trace validity. Investigations into infection profiling further emphasize the importance of maintaining consistent data structures for reliable system behavior [12].

The cumulative effect of these factors highlights the limitations of existing approaches to execution trace analysis. Research on antimicrobial activity demonstrates how system variability can influence outcomes under changing conditions [13], while broader analyses of resistance patterns emphasize the need for adaptive and resilient system design [14]. This research addresses these challenges by proposing a structured execution trace forensic framework for low-code platforms, enabling improved anomaly detection, compliance assurance, and system transparency.

2. Methodology

The proposed methodology establishes a formal framework for execution trace forensics in low-code platforms by modeling workflow execution as a structured sequence of state transitions. Each workflow instance is represented as a directed execution graph $G = (V, E)$, where nodes Vcorrespond to execution states and edges Edenote transitions triggered by events, conditions, or data transformations. The execution trace is defined as an ordered sequence $T = \{s_1, s_2, \dots, s_n\}$, capturing the temporal progression of states. Forensic analysis aims to reconstruct this sequence and identify deviations from expected execution paths under compliance constraints.

To quantify deviations, a baseline execution model $T_0$is constructed from validated workflow runs under controlled conditions. Any observed trace $T'$is compared against T_0using a deviation function:

$$\Delta(T', T_0) = \sum_{i=1}^{n} \delta(s_i', s_i),$$

where $\delta(\cdot)$measures state-level divergence based on transition mismatches, missing events, or unexpected branching. This formulation enables detection of both structural and temporal anomalies, providing a foundation for identifying compliance violations and execution inconsistencies.

The methodology categorizes anomalies into distinct classes based on their origin and propagation behavior. Structural anomalies arise from schema mismatches, missing attributes, or incompatible data formats, while semantic anomalies result from incorrect interpretation of data or transformation rules. Temporal anomalies capture delays, reordering, or duplication of events, and control-flow anomalies represent deviations in execution logic such as unauthorized branching or skipped steps. These categories provide a comprehensive taxonomy for forensic evaluation and are systematically defined in Table 1.

The forensic evaluation process operates across multiple layers of the low-code execution pipeline. At the data layer, schema validation and consistency checks are applied to detect structural anomalies. At the transformation layer, mapping rules and expression evaluations are analyzed to identify semantic inconsistencies. Temporal analysis focuses on event sequencing and latency patterns, while control-flow verification ensures adherence to predefined workflow logic. This layered approach enables precise localization of anomalies and supports comprehensive forensic reconstruction.

To capture temporal behavior, the methodology incorporates time-indexed analysis of execution traces. Each state transition is associated with a timestamp t_i, enabling the construction of a temporal profile $\mathcal{T}(T) = \{(s_i, t_i)\}$. Temporal anomalies are detected by comparing observed timing patterns against baseline expectations using:

$$\Theta(T') = \sum_{i=1}^{n} | t_i' - t_i |,$$

Table 1. Execution Trace Anomaly Categories and Forensic Evaluation Metrics

Anomaly Category	Description	Affected Layer	Forensic Metric
Structural Anomaly	Schema mismatch, missing fields, data format inconsistencies	Data Layer	Schema Divergence Score
Semantic Anomaly	Incorrect data interpretation or transformation	Transformation Layer	Semantic Consistency Index
Temporal Anomaly	Event delay, duplication, or reordering	Execution Timeline	Temporal Drift Measure
Control-Flow Anomaly	Unexpected branching or skipped workflow steps	Logic Layer	Path Deviation Score
Integration Anomaly	Failure in external service interaction or API inconsistency	Integration Layer	External Call Failure Rate

which quantifies cumulative temporal deviation. This metric is particularly useful in compliance scenarios where timing constraints, such as approval delays or transaction processing windows, are critical.

Control-flow verification is performed using path consistency analysis, where execution paths are mapped against a predefined workflow graph. Let $P(T')$denote the path extracted from the observed trace and P_0represent the expected path. The deviation is measured as:

$$\Gamma(T') = \text{dist}(P(T'), P_0),$$

where $\text{dist}(\cdot)$represents a graph-based distance metric. This enables identification of unauthorized transitions, skipped states, and redundant execution loops, which are key indicators of compliance violations.

Integration-level anomalies are analyzed through correlation of trace events across distributed components. Each external interaction is logged with identifiers linking it to internal execution states. A correlation function $\mathcal{C}(T')$maps internal and external traces to detect inconsistencies such as missing responses or mismatched data exchanges. This is essential for reconstructing end-to-end workflows that span multiple systems, particularly in enterprise environments with heterogeneous architectures.

Finally, the methodology includes a composite forensic score that aggregates multiple anomaly metrics to provide an overall assessment of execution integrity:

$$\Phi(T') = \alpha\Delta(T') + \beta\Theta(T') + \gamma\Gamma(T') + \lambda\mathcal{C}(T'),$$

where $\alpha, \beta, \gamma, \lambda$are weighting coefficients. This unified score enables ranking of execution traces based on severity of deviation and supports automated prioritization of forensic investigations. The proposed framework thus provides a rigorous, multi-layered approach for analyzing execution trace anomalies in low-code platforms, ensuring compliance, transparency, and system reliability.

3. Results and Discussion

The forensic evaluation reveals that execution trace deviations in low-code platforms exhibit both stage-wise amplification and anomaly-type differentiation, rather than a simple linear increase. Initial workflow stages such as data ingestion and validation maintain relatively low deviation levels due to structured input constraints and predefined schema checks. However, as execution progresses into transformation and orchestration layers, deviations begin to diversify across anomaly types, indicating that different categories of inconsistencies emerge depending on the computational context of each stage.

The multidimensional behavior of execution trace deviations is illustrated in Figure 1, which presents a heatmap capturing anomaly intensity across workflow stages and anomaly categories. The visualization shows that structural anomalies dominate early stages, while semantic and control-flow anomalies intensify during transformation and orchestration phases. Temporal and integration anomalies become more prominent in later stages, particularly during external system interactions. This distribution confirms that execution trace deviations are not uniform but are context-sensitive and stage-dependent, requiring targeted forensic strategies.

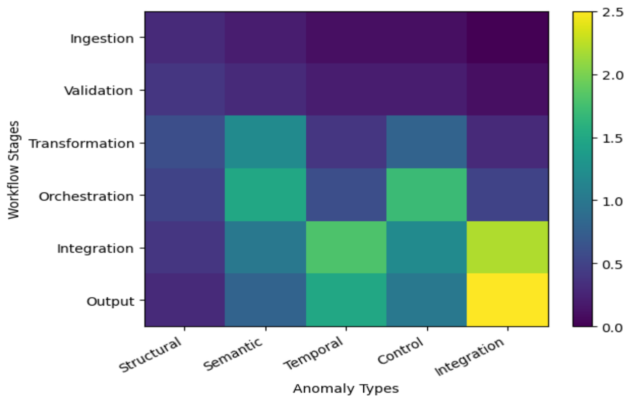

Fig. 1. Heatmap of Execution Trace Deviation Across Workflow Stages and Anomaly Types

A deeper analysis indicates that transformation layers act as the primary source of semantic inconsistencies. These arise due to complex mapping rules, conditional evaluations, and intermediate data transformations that are dynamically generated in low-code environments. The heatmap clearly shows higher intensity values for semantic anomalies in these stages, suggesting that even minor inconsistencies in transformation logic can significantly distort execution traces. This aligns with the theoretical expectation that intermediate representations are critical points of failure propagation.

Control-flow anomalies exhibit a different pattern, with peak intensity observed in orchestration stages where workflow branching and event-driven triggers are executed. These anomalies are particularly challenging to detect using conventional logging approaches, as they often involve valid but unintended execution paths. The heatmap highlights this behavior by showing localized high-intensity regions for control-flow deviations, indicating the need for graph-based path validation techniques in forensic analysis.

Integration stages demonstrate the highest overall deviation intensity, primarily due to interactions with external systems and APIs. Variability in response formats, latency, and data consistency leads to integration anomalies that disrupt trace continuity. The heatmap reveals strong coupling between temporal and integration anomalies in these stages, emphasizing the importance of cross-system trace correlation mechanisms. Without such mechanisms, forensic reconstruction remains incomplete and potentially misleading.

4. Conclusion

This study presented a structured forensic framework for analyzing execution trace deviations in AI-assisted low-code platforms operating under compliance-critical conditions. The results demonstrate that execution traces cannot be treated as simple linear logs but must be interpreted as multi-dimensional artifacts influenced by workflow stages, transformation logic, and integration behavior. By modeling execution as a state-transition system and introducing anomaly-specific metrics, the framework enables systematic identification of deviations that are otherwise hidden within automated execution pipelines.

The findings reveal that execution trace deviations exhibit strong stage dependency, with transformation and orchestration layers acting as primary sources of semantic and control-flow anomalies, while integration layers contribute significantly to temporal and cross-system inconsistencies. The heatmap-based analysis confirms that different anomaly types dominate at different stages, indicating that forensic evaluation must be context-aware rather than uniform. This insight is critical for compliance validation, where understanding the origin and propagation of deviations is essential for ensuring system integrity.

Another key contribution of this work is the demonstration that conventional logging and monitoring approaches are insufficient for forensic-grade analysis. While logs capture events, they do not provide structured mechanisms for reconstructing execution causality or detecting subtle inconsistencies across layers. The proposed framework addresses this limitation by integrating structural, temporal, and control-flow analysis into a unified model, enabling comprehensive trace reconstruction and anomaly classification in complex low-code environments.

In conclusion, the study establishes that execution trace forensics is a fundamental requirement for ensuring reliability, transparency, and compliance in AI-driven low-code systems. Future research should focus on developing automated forensic engines capable of real-time anomaly detection, integrating machine learning for adaptive trace interpretation, and establishing standardized benchmarks for evaluating forensic accuracy. Such advancements will be essential for enabling trustworthy deployment of low-code platforms in highly regulated domains.

References

1. Manzoor, M., Maziz, M. N. H., Subrimanyan, V., Shirin, L., Doustjalali, S. R., Sabet, N. S., ... & Mathialagan, A. (2022). Attitudes towards and the confidence in acceptance of telemedicine among the people in Sabah, Malaysia. *International Journal of Health Sciences*, *6*(S3), 2376-2386.
2. Vijayakumar, K., Maziz, M. N. H., & Prabha, S. (2025, March). Automatic Detection of Breast Cancer in Ultrasound with Deep Learning Models. In *2025 International Conference on Frontier Technologies and Solutions (ICFTS)* (pp. 1-6). IEEE.
3. MKK, F., Rashid, S. S., MHM, N., Baharudin, R., & Ramli, A. N. M. (2019). A clinical update on Antibiotic Resistance Gram-negative bacteria in Malaysia-a review. *arXiv preprint arXiv:1903.03486*.
4. Tien, L. P., Atiqah, N., Vytialingam, N., MA, R., Kabir, M. S., Shirin, L., ... & MHM, N. (2022). STRESS AND QUALITY OF LIFE AMONG FATHERS OF SPECIAL NEEDS CHILDREN IN KLANG VALLEY. *Journal of Pharmaceutical Negative Results*, *13*.

5. Nazmul, M. H. M., Jamal, H., & Fazlul, M. K. K. (2012). Acinetobacter species-associated infections and their antibiotic susceptibility profiles in Malaysia. *Biomed Res-India*, *23*(4), 571-575.
6. Vijayakumar, K., Maziz, M. N. H., Ramadasan, S., Prabha, S., & Kumaar, K. S. N. (2024, May). Automatic classification of healthy/TB chest X-ray using DeepLearning. In *2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)* (pp. 1-5). IEEE.
7. HasanMaziz, V. V. S. S., Ragavan, N. D., Arvind, C., Vairavan, S., &Neevashini, C. (2023). GC-Ms Analysis and Antibacterial Activity of Dryopteris Hirtipes (Blumze) Kuntze Linn. *Journal of Survey in Fisheries Sciences*, *10*(1S), 3718-3726.
8. Ismail, S., Radu, S., Sidek, K., Arifffin, M. A., Maziz, M. N. H., Hamzah, I., & Abdulla, M. A. (2003). Effect of dexamethasone treatment on the hematological and histological parameters of mice following experimental bacterial infection. *J Anim Vet Adv*, 2, 231-236.
9. Velmurugan, C., Subramaniyan, V., Ilanthalir, S., Fuloria, S., Sekar, M., Fuloria, N. K., & Hasan Maziz, M. N. (2022). Evaluation of anti-diabetic and wound healing potential of Ethiopia plant'Ruta graveolens' in diabetic induced rat.
10. Vijayakumar, K., Maziz, M. N. H., Ramadasan, S., Balaji, G., & Prabha, S. (2024, May). Benign/Malignant Skin Melanoma Detection from Dermoscopy Images using Lightweight Deep Transfer Learning. In *2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI)* (pp. 1-5). IEEE.
11. Maziz, M. N. H., Fazlul, M. K. K., Deepthi, S., Munirah, B., Farzana, Y., Najnin, A., & Srikumar, C. (2019). A study of comparison on knowledge and misconceptions about Hiv/Aids among students in a private university In Malaysia. *Malaysian Journal of Public Health Medicine*, *19*(1), 134-142.
12. Nazmul, M. H. M., Salmah, I., Jamal, H., & Ansary, A. (2008). Molecular characterization of verotoxin gene in enteropathogenic Escherichia coli isolated from Miri Hospital, Sarawak, Malaysia. *Biomed. Res*, *19*(1), 9-12.
13. MKK, F., MA, R., & MHM, N. (2019). Detection of CTX-M-type ESBLs from Escherichia coli clinical isolates from a tertiary hospital, Malaysia. *Baghdad Science Journal*, *16*(3), 20.
14. Mkk, F., Sp, D., & Irfan, M. (2019). Antibacterial and antifungal activity of various extracts of Bacopa monnieri. *arXiv preprint arXiv:1909.01856*.