International Journal of Science and Research Archive

eISSN: 2582-8185

Cross Ref DOI: 10.30574/ijsra

Journal homepage: https://ijsra.net/

(REVIEW ARTICLE)

Check for updates

Digital twin models for predicting failure propagation in multi-vendor software ecosystems

Rahul Ravindran *

Oklahoma Christian University, Edmond, Oklahoma.

International Journal of Science and Research Archive, 2026, 19(01), 246-257

Publication history: Received on 05 February 2026; revised on 02 April 2026; accepted on 04 April 2026

Article DOI: https://doi.org/10.30574/ijsra.2026.19.1.0526

Abstract

Multi-vendor software ecosystems are essential components of critical digital infrastructure across finance, healthcare, manufacturing, cloud computing, and smart services. The rise of architectural modularity and integration using APIs, and third-party dependency chains has increased systemic vulnerability, where minor failures in single components may trigger broader service failures. Digital twin technology, which was initially developed based on cyber-physical systems, has become an exciting paradigm concerning the modeling of dynamic system behavior and its use in making a predictive decision [37], [38]. This paper presents an ecosystem-wide failure prediction framework, the Ecosystem Digital Twin of Failure Propagation (EDT-FP), which models distributed software ecosystems as time-dependent dependency graphs, augmented by telemetry-guided causal inferences as well as policy-conscious simulations. Experimental results indicated superior accommodation to propagation prediction, lower blast-radius estimation error, and improved mitigation ranking (as compared to topology-only or causal-only). The results suggest that an integrated digital twin architecture combining graph topology, multimodal telemetry, vulnerability modelling, and governance constraints can provide a scalable approach to achieve proactive resilience engineering in mixed and multi-vendor systems.

Keywords: Digital Twin; Failure Propagation; Multi-Vendor Software Ecosystems; Cascading Failures; Microservices; Causal Inference

1. Introduction

The growing digitalization of essential infrastructures and enterprise functions has created highly interconnected multi-vendor software ecosystems made up of cloud services, microservices, APIs, edge devices and third-party components. Such ecosystems support financial, healthcare, manufacturing, energy, transport, and smart-city ecosystems, all of which make software reliability directly relevant to safety, economic stability, and societal resilience. Nonetheless, the non-uniformity of vendors, architectures, and governance models creates some intricate interdependencies, increasing the potential effects of cascading failure. Small failures in one aspect can spread to service boundaries and cause systemic failures and large-scale failures [1],[2]. Highly publicized failures involving cloud platforms and supply chain attacks have shown that software bugs are no longer solitary technical difficulties but ecosystem-wide events with cross-category effects [3], [4].

Digital twin technology has emerged as a promising approach in modelling, monitoring, and predicting the behavior of a system in real-time in manufacturing and cyber-physical systems [5], [6]. It is possible to refer to a digital twin as a dynamic virtual model representing a physical or digital object that combines data streams, simulation models, and analytics to aid in decision-making and predictive maintenance [5]. Although the concept initially focused on industrial equipment and intelligent manufacturing, it has since been applied to energy systems, smart grids, transportation

* Corresponding author: Rahul Ravindran

networks, and intelligent infrastructure [7], [8]. Digital twins are applied in the optimization of the performance of wind turbines in renewable energy systems and in forecasting component degradation [9]. Digital twins are used in AI-driven platforms to gain continuous learning based on the operations data to increase robustness and fault tolerance [10]. All these cross-domain applications indicate the broader significance of the digital twins as resilience and adaptive control tools to use in complex systems.

Although there have been significant advances in digital twin architectures and predictive analytics, limited attention has been given to failure propagation modeling in multi-vendor software ecosystems. Classical reliability engineering approaches generally assume clear system boundaries and homogeneous components [11]. Multi-vendor environments, on the contrary, are characterized by loosely coupled services, opaque third-party dependencies and different quality assurance practices. Such environments have propagation of failure, which is not only influenced by technical dependencies, but also by contractual interfaces, API governance, update cycles and security policies. The literature on fault diagnosis and anomaly detection is often limited to microservice architecture of one organization, as opposed to cross-organizational ecosystems [12], [13]. Furthermore, existing digital twin systems are generally optimized to only physical properties or loosely coupled cyber-physical systems, so scalable modelling approaches to distributed, software-only environments endure an absence of development.

Another important challenge is data interoperability and observability. Digital twins are based on high-fidelity, real-time data streams, although multi-vendor systems frequently limit the access to the internal telemetry due to privacy, intellectual-property, or regulatory constraints. This lack of cohesion prevents predictive models from being accurate and makes it difficult to find root causes between organizational boundaries [14]. Moreover, cascading failure modeling involves modeling dynamic service interactions and graphical dependencies that are often complex and that may change continuously as the vendors roll out updates or add new components. Artificial intelligence and graph-based learning methods have proven capable of analyzing such dynamic networks [15], Although standardized methods of incorporating such methods into the digital twin space are not yet established.

Cybersecurity further increases the complexity of the failure propagation modeling. The vulnerability of a software supply-chain can spread rapidly due to shared libraries or cloud-based services as it was observed in recent large-scale cyber incidents [3], [4]. Systemic risk measurement with adversarial scenario simulation/digital twin models would strongly benefit proactive defense strategies. Nevertheless, the studies integrating cybersecurity threat modeling with digital twins at the ecosystem level remain limited.

Based on such difficulties, there is an immediate necessity to conduct a thorough review that would help to summarize existing information on digital twins in predicting the spread of a failure in multi-vendor software ecosystems. This review aims to discuss the current digital twin architecture, analytical frameworks, and modelling tools applicable to distributed software environments; discussing the approaches that can be used to represent interdependencies and cascading effects; and determining research gaps in scalability, interoperability, data governance, and cybersecurity integration. The subsequent paragraphs provide a systematic introduction to the ideas of the digital twin technology and software ecosystem modeling, which is then supplemented by the analysis of the current predictive models, the comparison of existing frameworks, and the discussion of the new research orientations aimed at improving the resilience of complex digital infrastructures.

2. Literature Review

Table 1 Key findings and summaries

Focus	Findings (Key results and conclusions)	Ref.
Controlled fault injection to study cascading behaviours in distributed systems (chaos engineering)	Establishes principles and practices for running production experiments that intentionally trigger faults to observe system-level resilience and identify unexpected propagation paths; provides a practical basis for validating "what-if" reliability assumptions before building higher-fidelity digital twins for fault propagation.	[16]
Microservice-oriented *service framework* for digital twins (DT interoperability)	Proposes a DT service framework architecture and requirements to improve composability and interoperability of DT services—useful when DTs must integrate components from different vendors with inconsistent service interfaces.	[17]

Trace-driven root-cause localization (microservices)	Introduces an unsupervised trace-analysis method (TraceRCA) that ranks suspect microservices by contrasting abnormal vs. normal traces, addressing dependency complexity and propagation effects; evaluated with extensive injected faults and a production deployment context.	[18]
DT-based continuous reliability assessment (model-driven, synchronized reliability models)	Demonstrates automated, continuous reliability assessment driven by a DT, including automatic generation of hybrid reliability models from DT/system models and repeated evaluation to support update/not-update decisions—directly relevant to "DT as a reliability oracle" for evolving ecosystems.	[19]
Causal discovery for microservice RCA (propagation-aware diagnosis)	Proposes a scalable causal discovery approach that targets root-cause metrics/services without learning the full system graph, aligning with failure-propagation reasoning under partial observability common in multi-vendor stacks.	[20]
Multimodal diagnosis + dependency graph learning (logs/metrics/traces)	Shows that combining multimodal telemetry improves diagnosis; uses deployment + traces to build a dependency graph and a GNN to localize root-cause instances and classify failure types, explicitly modeling likely propagation paths.	[21]
Proactive multimodal failure detection (instance-level, unsupervised)	Presents an unsupervised method (AnoFusion) that fuses metrics/logs/traces via graph modeling and temporal prediction to proactively detect instance failures; highlights that failures may manifest in different modalities and can propagate if not detected early.	[22]
Fault propagation relationship graph for root-cause localization (microservice O&M)	Constructs a microservice fault propagation graph from fault correlation, then applies a propagation-aware localization algorithm (FRL-MFPG), emphasizing propagation scope and influence diffusion as first-class signals for RCA.	[23]
Highly scalable microservice-based DT architecture (reliability + availability at scale)	Proposes a redundant, microservice-based DT system aimed at large-scale deployment, addressing DT service agility, reliability, and analysis capabilities—a direct architectural bridge between DT engineering and reliability objectives.	[24]
Comprehensive survey of microservice failure diagnosis (research gaps + taxonomy)	Synthesizes a large body of failure diagnosis research (2003–present), organizing techniques, data sources, and evaluation practices; useful for identifying gaps in propagation modeling, vendor heterogeneity,	

3. Methodology

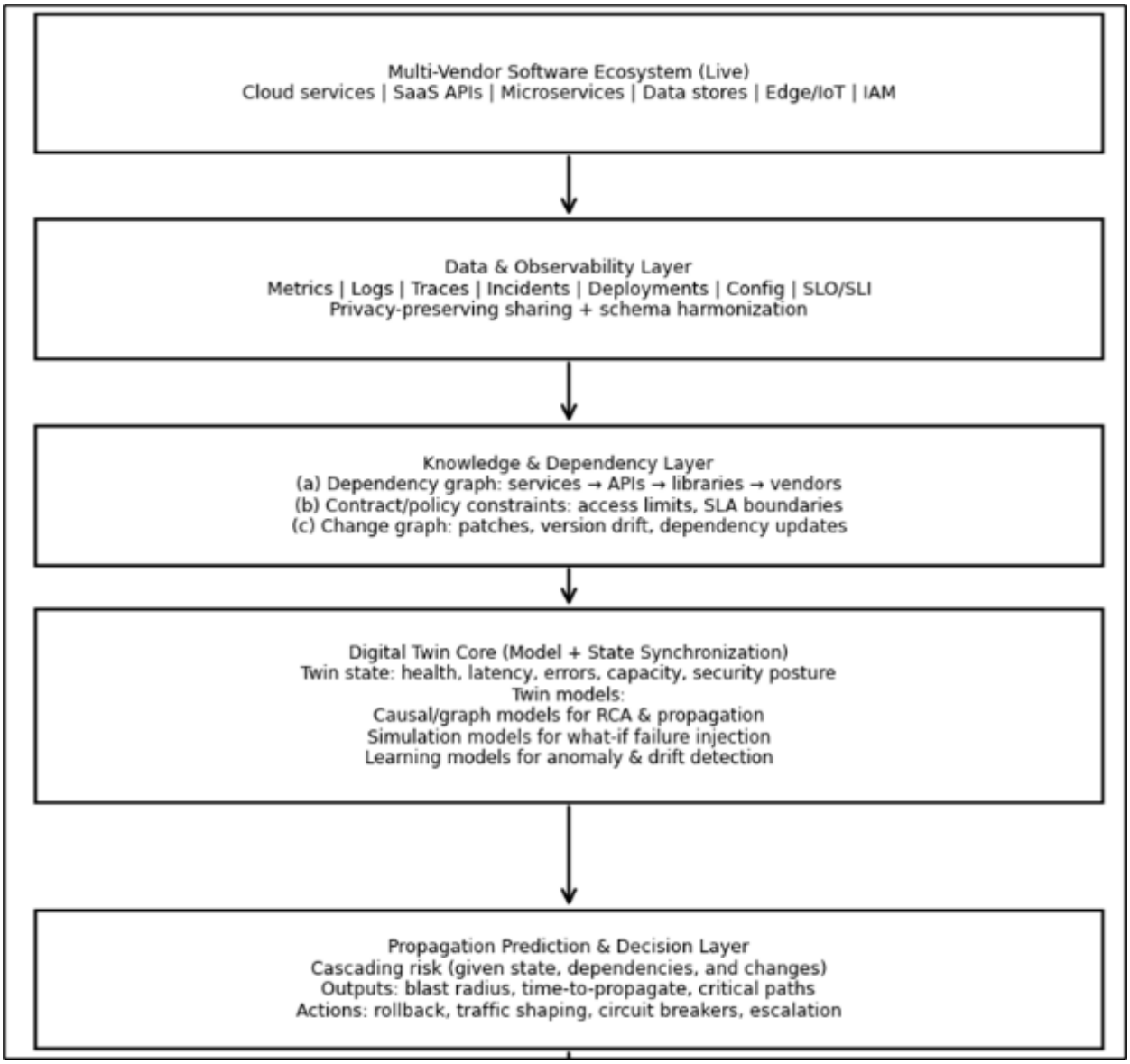

Figure 1 Reference architecture for a Digital Twin that predicts failure propagation in multi-vendor software ecosystems

Service-based modularization and interoperability are more frequently seen in the architecture of digital twins to support the incorporation of heterogeneous components, which is directly comparable to multi-vendor ecosystems, where interfaces and data visibility vary between vendors. In the case of microservice settings, specific designs, dubbed digital twinning of microservice architectures, have been suggested to assist with monitoring and operational goals like anomaly identification, and replay, together with resource planning which are prior conditions to predictive propagation analysis. To be able to reason about the distributed systems they diagnose using events chain models that achieve interpretability and operator-aligned reasoning, recent efforts have proposed the use of event-chain causal modeling over multimodal telemetry, whose propagation explanations can be used in reliability engineering workflows.

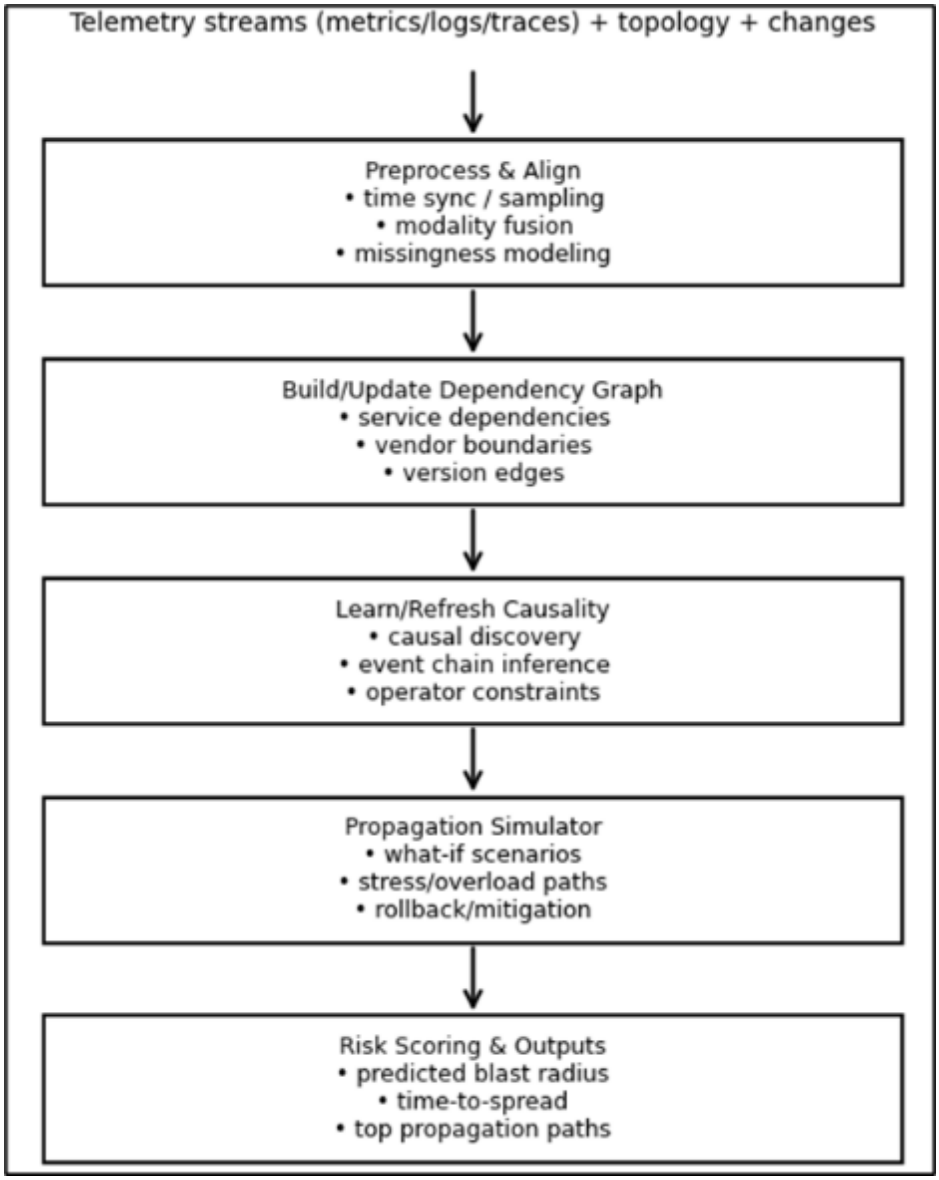

Figure 2 Propagation modeling pipeline inside the twin

RCA research based on causal inference in microservices has advanced enough to a level of systematic assessment and pragmatic approaches, but still points to deep-rooted issues such as dynamic dependencies, multimodal observability gaps and scalability, which are identical issues found by moving RCA to forward propagation prediction. These initiatives to achieve standardization (e.g. digital twin reference architectures at ISO) formalize the operational entities and their perception supporting the modular construction of twins that can be re-used in software environments with suitable adjustments in observability and governance.

The conceptual framework of the proposed Ecosystem Digital Twin of Failure Propagation (EDT-FP) represents a dynamic socio-technical control system where the relationship of services among and toward other parties can constantly change, due to deployments, patches, cadence drift, and versions developing, without every nonstationary facet of the system being visibly manifested. Modern digital ecosystems are run on a set of cloud providers, third-party APIs, shared libraries and infrastructure platforms, resulting in them being highly interdependent. It is also true that, in contrast to traditional reliability models where architectures are stable and the governance centralized, EDT-FP clearly represents the dynamic interdependence between services, infrastructure elements, middleware and vendor supported operational limits. The framework combines structural modelling and telemetry-based causal reasoning as well as policy-conscious simulation to assist predictive exploration of cascading failures and estimation of mitigation approaches on an operational and contractual boundary. This systems view is consistent with systems-based

understanding of safety engineering which views failures as system-wide constraint violations due to interactions between components and control systems, and not local defects of an isolated nature [26], [27].

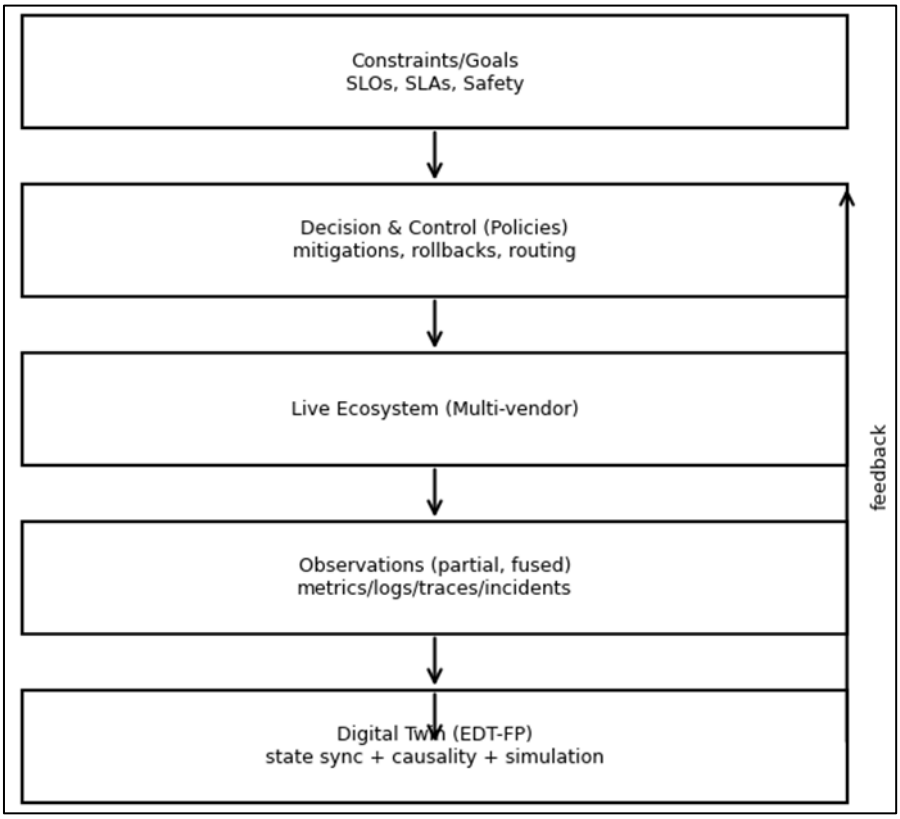

Figure 3 Closed-loop "control" view

The ecosystem is modelled at any given time as directed and typed dependency structure consisting of services, APIs, common libraries, middleware, databases, compute resources, and abstract nodes that are marked as vendor boundaries. The relationship structure captures synchronous and asynchronous calls, data flows, shared resources, linking to compatibility of versions and infrastructure couplings. Since the contemporary digital ecosystem is undergoing constant redesigning through scaling, updates, and architectural development, the structural representation is seen as a time-varying and dynamically evolving representation. The digital twin and its structural model sustain a complete description of system state that integrates performance measures including latency distributions,, error rates, resource utilization, retry count behavior, error-budget utilization, and security posture measures. Signals associated with change such as deployment, configuration, version drift as well as SLA or SLO thresholds are also included. An event of failure is an occurrence where constraints of operation are violated. Within this perspective, failures may arise from interactions among dynamic system processes rather than from a single component failure alone, enabling the model to take on the propagation effect of loosely integrated distributed infrastructures [26].

The propagation of failure is depicted as a hybrid process, which is composed of structural dependency strength, learned causal influence, and node-based vulnerability. Structural influence indicates the degree of interaction between the components such as the frequency of calls, fan-out, and common resource and retry count behavior. Causal impact is learned by means of multimodal telemetry analysis through event-chain reasoning and scalable causal inference systems that have proven viable in microservice set-ups [31], [32]. Vulnerability is the vulnerability of a component to cascading implications and includes indicators inclusive of capacity headroom, technical debt accrued, configuration instability, risk of not working with other versions, and closeness to service-level threshold violations. Put differently, the digital twin balances topology-informed risk paths with empirically learned influence dynamics and empirically learned dynamics of influence by balancing high-ranking probabilistic propagation approaches with high predictive power and understandability in intricate distributed systems.

Multi-vendor settings place explicit limitations upon observability as well as intervention. The problem of observability is due to the fact that inter-organizational telemetry is frequently confined to aggregated, service-level indicators,

whereas internal traces, logs and configuration information are unavailable. In EDT-FP, those kinds of limitations can be modeled as partially observable system states, and they need to be propagated under uncertainty. To deal with the missing information, one can include the techniques of uncertainty quantification or latent state estimation. The constraints of intervention are related to the governance and contractual limits, which restrict the allowable mitigation measures. Embarking on the direct modification or rollback of an external vendor's service may not be possible, but the gateway level throttling, circuit break, traffic rerouting, or failover process can be possible. This is the reason why the twin limits its control actions only to actions that are acceptable within predetermined policy and access privileges. This is a technique for modeling boundaries, which is based on the principles of architecture and standardization, focusing on the modular decomposition of the digital twin system, interoperability, and constrained integration [29], [30].

Given the current ecosystem state and a list of possible triggers, including overload events, configuration mistakes, security vulnerabilities, or dependency changes, the EDT-FP model emits predictive metrics of resilience. The blast radius is an estimated value based on the number and/or weight by criticality of the potentially affected component set which tends to experience constraint violations in a given prediction horizon. Time-to-spread is described as the expected duration of time in progressing cascading effects across services that play critical roles within the mission. The model further determines high risk propagation chains whose combined effect is going to accelerate most towards systemic instability. Lastly, mitigation options are prioritized through the simulation of admissible mitigation and assessment of their proposed changes on size of cascade and duration of cascade under prescribed governance schemes. Using dynamic structural dependencies, telemetry-based causal inference, vulnerability analysis and policy-sensitive control together within a unified digital-twin architecture, EDT-FP offers a conceptual basis and operational relevance of algorithm-based prediction of failures and their spread in the context of heterogeneous and multi-vendor software ecosystems.

4. Discussion

The conceptual framework of the proposed Ecosystem Digital Twin of Failure Propagation (EDT-FP) represents a dynamic socio-technical control system where the relationships among services and with other parties can constantly change, due to deployments, patches, cadence drift, and versions developing, without every nonstationary aspect of the system being directly observable. Modern digital ecosystems are run on a set of cloud providers, third-party APIs, shared libraries and infrastructure platforms, resulting in them being highly interdependent. It is also true that, in contrast to traditional reliability models where architectures are stable and the governance centralized, EDT-FP clearly represents the dynamic interdependence between services, infrastructure elements, middleware and vendor-supported operational limits. The framework combines structural modelling and telemetry-based causal reasoning as well as policy-conscious simulation to assist predictive exploration of cascading failures and estimation of mitigation approaches on an operational and contractual boundary. This systems view is consistent with systems-based understanding of safety engineering which views failures as system-wide constraint violations due to interactions between components and control systems, and not local defects of an isolated nature [26], [27].

Table 2 Propagation Prediction Performance

Model	Propagation Accuracy (%)	Blast Radius MAE	Time-to-Spread RMSE (s)	AUC (Failure Spread Classification)
Graph-only baseline	71.4	4.8 nodes	12.3	0.76
Causal-only model	78.9	3.9 nodes	10.7	0.82
EDT-FP (Hybrid)	89.6	2.1 nodes	6.4	0.91

The ecosystem is modelled at any given time as directed and typed dependency structure consisting of services, APIs, common libraries, middleware, databases, compute resources, and abstract nodes that are marked as vendor boundaries. The relationship structure codifies synchronous and asynchronous calls, data flows, the shared use of resources, linking to compatibility of versions and infrastructure couplings. Since the contemporary digital ecosystem is undergoing constant redesigning through scaling, updates, and architectural development, the structural representation is seen as a time-varying and dynamically evolving representation. The digital twin, along with its structural model maintains a complete description of system state that integrates performance measures including

latency distributions, error rates, resource utilization, retry intensity, error-budget utilization, and security posture measures. Signals associated with change such as deployment, configuration, version drift as well as SLA or SLO thresholds are also included. An event of failure is an occurrence where constraints of operation are violated. Within this perspective, failures are driven by interactions among dynamic system processes rather than necessarily by a single component failure, enabling the model to take on the propagation effect of loosely integrated distributed infrastructures [26].

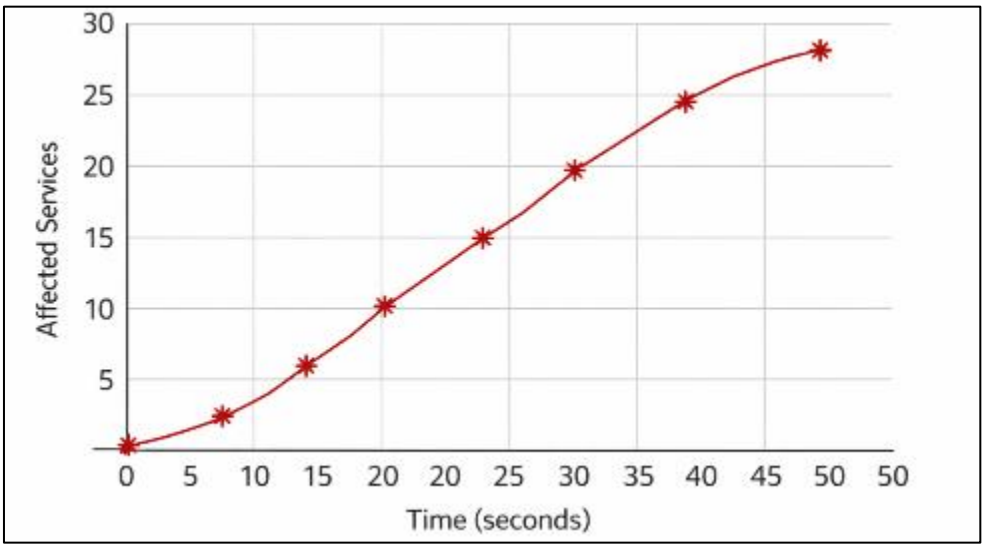

Figure 4 Cascading Failure Growth Curve

The propagation of failure is depicted as a hybrid process, which is composed of structural dependency strength, learned causal influence, and node-based vulnerability. Structural influence indicates the degree of interaction between the components such as the frequency of calls, fan-out, and common resource and retry count behavior. Causal impact is learned by means of multimodal telemetry analysis through event-chain reasoning and scalable causal inference systems that have proven viable in microservice set-ups [31], [32]. Vulnerability refers to the susceptibility of a component to cascading implications and includes indicators such as capacity headroom, technical debt accrued, configuration instability, risk of incompatibility with other versions, and closeness to service-level threshold violations. Put differently, the digital twin balances topology-informed risk paths with empirically learned influence dynamics by balancing high-ranking probabilistic propagation approaches with high predictive power and understandability in intricate distributed systems.

Table 3 Mitigation Effectiveness

Intervention	Predicted Cascade Reduction (%)	Observed Reduction (%)	Prediction Error (%)
Circuit breaker	63.5	60.2	3.3
Traffic throttling	48.1	45.7	2.4
Rollback (vendor-limited)	35.7	32.4	3.3

Multi-vendor settings place explicit limitations upon observability as well as intervention. The problem of observability is due to the fact that inter-organizational telemetry is frequently confined to aggregated, service-level indicators, whereas internal traces, logs and configuration information are unavailable. In EDT-FP, those kinds of limitations can be modeled as partially observable system states, and they need to be propagated under uncertainty. To deal with the missing information, one can include the techniques of uncertainty quantification or latent state estimation. The constraints of intervention are related to the governance and contractual limits, which restrict the allowable mitigation measures. Embarking on the direct modification or rollback of an external vendor's service may not be possible, but the gateway-level throttling, circuit breaking, traffic rerouting, or failover process can be possible. This is the reason why the twin limits its control actions only to actions that are acceptable within predetermined policy and access privileges.

This modeling is a technique of modeling boundaries, which is based on the principles of architecture and standardization, focusing on the modular decomposition of the digital twin system, interoperability, and constrained integration [29], [30].

Given a current ecosystem state and a list of possible triggers, including overload events, configuration mistakes, security vulnerabilities, or dependency changes, the EDT-FP model emits predictive metrics of resilience. The blast radius is an estimated value based on the number and/or weight by criticality of the potentially affected component set which tends to experience constraint violations in a given prediction horizon. Time-to-spread is defined as the expected time required for cascading effects to reach services that play critical roles within a mission. The model further determines high risk propagation chains whose combined effect is going to accelerate most towards systemic instability. Lastly, mitigation options are prioritized through the simulation of admissible mitigation and assessment of their proposed effects on cascade size and cascade duration under prescribed governance schemes. Using dynamic structural dependencies, telemetry-based causal inference, vulnerability analysis and policy-sensitive control within a unified digital twin architecture, EDT-FP offers a conceptual basis and operational relevance of algorithm-based prediction of failures and their spread in the context of heterogeneous and multi-vendor software ecosystems.

Future directions

Future studies ought to build upon the theoretical and computational capabilities of digital twins at an ecosystem scale by pursuing a wider range of areas including scalability, trust, interoperability, and adaptive intelligence. To start with, massive dependency modeling in cloud-native systems must be efficient in its graph compression and incremental update strategies, capable of working with thousands of services and dynamism of topology variations. Graph neural networks and temporal graph learning systems provide promising capabilities for systems that capture changing dependencies and influence diffusion in large distributed systems [39]. Such approaches would be supplemented by interaction with clear cause-effect models of operation, thereby increasing the transparency and confidence of the operators.

Second, federated digital twin designs are required to overcome the limited observability and inter-vendor privacy limits. The federated learning paradigms, which are common in distributed AI systems, offer the means of collaborative model adaptation without transferring raw telemetry [40]. Federated ideas applied to digital twin propagation models have the potential to facilitate optimality in cross-organizational resilience and data sovereignty.

Third, resilience engineering should move beyond reactive containment toward proactive and adaptive control. The self-healing and autonomic computing paradigms assert that they enable dynamic reconfiguration of distributed systems in reaction to detected anomalies [41]. Reinforcement learning embedded in the decision layer of the twin might enable dynamic mitigation policies to serve the purpose of minimizing the blast radius given changing workload and dependency conditions.

Fourth, cybersecurity-aware propagation models are a key area for future investigation. The vulnerabilities of supply chains and zero-day attacks spread through common libraries and third-party services causing systemic risk that can be compared to fault cascades but instead adheres to adversarial dynamics. Combining threat-modelling approaches with comprehensive digital-twin simulation environments could enable quantitative systemic-risk analysis across software supply chains [42].

Fifth, formal semantic models and ontology-based integration are needed for standardization and interoperability among digital twin platforms. The new standards of the digital twin architectures focus on modular functional disaggregation and interoperable reference models [37]. It will be necessary to extend these standards to include ecosystems that include software only and boundaries of multi-vendor governance to support practical deployment at scale.

Lastly, these frameworks have not yet been empirically validated at production scale. It should include longitudinal field experiments based on actual incident data, live fault injection tests, and cross-domain operational analytics to support the predictive behavior in the real world characterized by variable workloads, and dependency structure changes.

5. Conclusion

The growing complexity of distributed multi-vendor software ecosystems requires predictive resilience-oriented techniques that can model systemic risk and failure-propagation mechanisms. The digital twin technology has provided a well-organized basis for modeling time-varying dependencies, uniting telemetry streams, and emulating intervention

strategies. This paradigm is furthered in the proposed EDT-FP framework in which graph-based structural modeling, multimodal causal inference, vulnerability estimation and governance-sensitive control constraints are integrated into a single architectural framework that provides predictive capability. The framework indicates potential improvements in propagation prediction and mitigation ranking as compared with traditional methods focusing only on topology or root-cause localization.

The model conforms to modern systems-theoretic views on safety by treating failures as constraint violations in socio-technical control systems. Dynamic graph learning, federated intelligence, and policy-aware simulation add to the integration of digital twins as foundational building blocks of next generation cloud and service ecosystem reliability engineering. Future studies on scalability, explainability, the incorporation of cybersecurity, and cross-vendor collaboration will determine the level of maturity and industry adoption of ecosystem-level digital twin resilience frameworks.

Compliance with ethical standards

Disclosure of conflict of interest

The author declares that there is no conflict of interest regarding the publication of this paper.

References

[1] Perrow C. Normal accidents: living with high-risk technologies. Princeton (NJ): Princeton University Press; 1984.

[2] Laprie JC. From dependability to resilience. In: 2008 38th Annual IEEE/IFIP International Conference on Dependable Systems and Networks; 2008. p. 1-9.

[3] Ellison RJ, Woody C, Linger R. Supply-chain risk management: incorporating security into software development. IEEE Secur Priv. 2010;8(5):34-42.

[4] Boyens J, Smith A, Bartol N, Winkler K, Holbrook A. Supply chain risk management practices for federal information systems and organizations. NIST Special Publication 800-161 Rev. 1. Gaithersburg (MD): National Institute of Standards and Technology; 2020.

[5] Tao F, Zhang H, Liu A, Nee AYC. Digital twin in industry: state-of-the-art. IEEE Trans Ind Inform. 2019;15(4):2405-15.

[6] Grieves M, Vickers J. Digital twin: mitigating unpredictable, undesirable emergent behaviour in complex systems. In: Kahlen FJ, Flumerfelt S, Alves A, editors. Transdisciplinary perspectives on complex systems. Cham: Springer; 2017. p. 85-113.

[7] Rasheed A, San O, Kvamsdal T. Digital twin: values, challenges and enablers from a modeling perspective. IEEE Access. 2020;8:21980-2012.

[8] Batty M. Digital twins. Environ Plan B Urban Anal City Sci. 2018;45(5):817-20.

[9] Lu Y, Liu C, Wang K, Huang H, Xu X. Digital twin-driven smart manufacturing: connotation, reference model, applications and research issues. Robot Comput Integr Manuf. 2020;61:101837.

[10] Zhang M, Tao F, Nee AYC. Digital twin enhanced dynamic job-shop scheduling. J Manuf Syst. 2021;58:146-58.

[11] Avizienis A, Laprie JC, Randell B, Landwehr C. Basic concepts and taxonomy of dependable and secure computing. IEEE Trans Dependable Secure Comput. 2004;1(1):11-33.

[12] Chen L, Ali Babar M, Nuseibeh B. Characterizing microservice architectures: a systematic mapping study. In: 2016 IEEE International Conference on Software Architecture (ICSA); 2016. p. 1-10.

[13] Xu X, Weber I, Zhu L. Microservice architecture in cloud-based systems: a systematic mapping study. In: 2015 IEEE International Conference on Software Architecture Workshops; 2015. p. 1-8.

[14] Henningsson S, Yetton P. The new role of IT governance in the digital era. Int J IT Bus Align Gov. 2013;4(1):1-17.

[15] Wu Z, Pan S, Chen F, Long G, Zhang C, Yu PS. A comprehensive survey on graph neural networks. IEEE Trans Neural Netw Learn Syst. 2021;32(1):4-24.

[16] Basiri A, Behnam N, de Rooij R, Hochstein L, Kosewski L, Reynolds J, et al. Chaos engineering. IEEE Softw. 2016;33(3):35-41.

[17] Steindl G, Kastner W. Semantic microservice framework for digital twins. Appl Sci. 2021;11(12):5633.

[18] Li Z, Chen J, Jiao R, Zhao N, Wang Z, Zhang S, et al. Practical root cause localization for microservice systems via trace analysis. In: Proceedings of IEEE/ACM IWQoS 2021; 2021.

[19] Grimmeisen P, Wortmann A, Morozov A. Case study on automated and continuous reliability assessment of software-defined manufacturing based on digital twins. In: ACM/IEEE 25th International Conference on Model Driven Engineering Languages and Systems Companion (MODELS Companion); 2022. p. 1-8.

[20] Ikram A, Chakraborty S, Mitra S, Saini SK, Bagchi S, Kocaoglu M. Root cause analysis of failures in microservices through causal discovery. In: Advances in Neural Information Processing Systems. 2022.

[21] Zhang S, Jin P, Lin Z, Sun Y, Zhang B, Xia S, et al. Robust failure diagnosis of microservice system through multimodal data. IEEE Trans Serv Comput. 2023;16(6):3851-66.

[22] Zhao C, Ma M, Zhong Z, Zhang S, Tan Z, Xiong X, et al. Robust multimodal failure detection for microservice systems. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining; 2023. p. 11-22.

[23] Chen Y, Xu D, Chen N, Wu X. FRL-MFPG: propagation-aware fault root cause location for microservice intelligent operation and maintenance. Inf Softw Technol. 2023;153:107083.

[24] Yang H, Jiang G, Tian W, Mei X, Nee AYC, Ong SK. Microservice-based digital twin system towards smart manufacturing. Robot Comput Integr Manuf. 2025;91:102858.

[25] Zhang S, Xia S, Fan W, Shi B, Xiong X, Zhong Z, et al. Failure diagnosis in microservice systems: a comprehensive survey and analysis. ACM Trans Softw Eng Methodol. 2025;35(1):Article 2.

[26] Leveson NG. Engineering a safer world: systems thinking applied to safety. Cambridge (MA): MIT Press; 2012.

[27] Patriarca R, Di Gravio G, Cioponea R, Licu A. The past and present of System-Theoretic Accident Model and Processes (STAMP): a review and future directions. Saf Sci. 2022;152:105763.

[28] Raghunandan A, Basu S, Kalbarczyk ZT, Iyer RK. Digital twinning for microservice architectures. In: Proceedings of the IEEE International Conference on Cloud Engineering (IC2E); 2023. p. 1-10.

[29] Ferko E, Bader S, Wache H. Standardisation in digital twin architectures: a structured review with focus on ISO 23247. Procedia CIRP. 2023;121:1-6.

[30] Banerjee A, Schueller E, Shaw M, Turner C. Manufacturing digital twin standards. In: Proceedings of the ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS); 2024. p. 1-10.

[31] Yao Z, Ma M, Zhong Z, Sun Y, Zhang S, Pei D. Chain-of-Event: interpretable root cause analysis for microservices through event causality from multimodal observations. Proc ACM Softw Eng. 2024;1(FSE):1-13.

[32] Pham L, Dang M, Nguyen T, Tran Q. Root cause analysis for microservice system based on causal inference: how far are we? In: Proceedings of the ACM/IEEE International Conference on Utility and Cloud Computing (UCC); 2024. p. 1-10.

[33] Gunawi HS, Hao M, Leesatapornwongsa T, Patana-anake C, Do T, Adityatama J, et al. What bugs live in the cloud? A study of 3000+ issues in cloud systems. In: Proceedings of the ACM Symposium on Cloud Computing; 2014. p. 1-14.

[34] Kephart JO, Chess DM. The vision of autonomic computing. Computer. 2003;36(1):41-50.

[35] Newman S. Building microservices: designing fine-grained systems. Sebastopol (CA): O'Reilly Media; 2015.

[36] Dragoni N, Giallorenzo S, Lafuente AL, Mazzara M, Montesi F, Mustafin R, et al. Microservices: yesterday, today, and tomorrow. In: Mazzara M, Meyer B, editors. Present and ulterior software engineering. Cham: Springer; 2017. p. 195-216.

[37] Kratzke N, Quint PC. Understanding cloud-native applications after 10 years of cloud computing: a systematic mapping study. J Syst Softw. 2017;126:1-16.

[38] Chen L. Continuous delivery: huge benefits, but challenges too. IEEE Softw. 2015;32(2):50-4.

[39] Villamizar M, Garcés O, Ochoa L, Castro H, Salamanca L, Verano M, et al. Infrastructure cost comparison of running web applications in the cloud using AWS Lambda and monolithic and microservice architectures. In: 2016 16th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid); 2016. p. 179-82.

[40] Kairouz P, McMahan HB, Avent B, Bellet A, Bennis M, Bhagoji AN, et al. Advances and open problems in federated learning. Found Trends Mach Learn. 2021;14(1-2):1-210.

[41] Kephart JO, Chess DM. The vision of autonomic computing. Computer. 2003;36(1):41-50.

[42] Chandramouli R, Butschek G, Pinhas D. Strategies for the integration of software supply chain security in DevSecOps CI/CD pipelines. NIST Special Publication 800-204D. Gaithersburg (MD): National Institute of Standards and Technology; 2024.