INTERNATIONAL JOURNAL OF ECONOMIC SOCIAL SCIENCE AND MANAGEMENT LAW
E-ISSN:3070-0558
VOL.4,NO.4 (2023)

OPENAPI-BASED INTEROPERABILITY FRAMEWORK FOR INTELLIGENT MANUFACTURING AND INDUSTRIAL DIGITAL TWIN PLATFORMS

Kang Min-Seok
Research Author

Abstract

The rapid development of intelligent manufacturing environments has increased the demand for seamless interoperability among heterogeneous industrial machines, Industrial Internet of Things devices, cyber-physical production systems, enterprise applications, cloud services, analytical engines, and Digital Twin platforms. Despite significant progress in Industry 4.0 technologies, many manufacturing organizations continue to operate fragmented software ecosystems in which machine data, process information, maintenance records, quality indicators, simulation models, and business applications are distributed across incompatible interfaces and proprietary communication structures. This paper proposes an OpenAPI-based interoperability framework for intelligent manufacturing and industrial Digital Twin platforms. The framework applies standardized API descriptions, reusable service contracts, machine-readable interface specifications, semantic data mapping, secure API gateways, event-driven integration, edge-cloud connectivity, Digital Twin synchronization, and intelligent analytical services to establish consistent communication across heterogeneous industrial environments. The proposed approach organizes interoperability into physical asset connectivity, Industrial Internet of Things data acquisition, edge processing, canonical information modeling, OpenAPI contract management, API gateway coordination, Digital Twin services, analytical intelligence, manufacturing application integration, and governance layers. OpenAPI specifications are used to describe endpoints, request structures, response schemas, authentication requirements, service versions, error conditions, and operational metadata so that manufacturing services can be discovered, validated, integrated, and maintained more systematically. The framework further supports real-time synchronization between physical industrial assets and their Digital Twin representations by standardizing interfaces for telemetry ingestion, state retrieval, event notification, simulation invocation, predictive maintenance, process optimization, and control recommendations. A representative analytical evaluation is presented to examine interoperability performance across integration time, API response latency, successful transaction rate, data consistency, Digital Twin synchronization delay, service reuse, fault recovery, and engineering effort. The results demonstrate that the proposed OpenAPI-centered approach can reduce integration complexity, improve interface consistency, accelerate service onboarding, and support scalable industrial Digital Twin ecosystems. The study concludes that standardized API contracts combined with semantic mapping, secure gateway management, event-driven communication, and Digital Twin orchestration provide a practical foundation for flexible, intelligent, and interoperable manufacturing systems.

Keywords: OpenAPI, Intelligent Manufacturing, Digital Twin, Industrial Interoperability, Industry 4.0, Industrial Internet of Things, API Gateway, Smart Manufacturing, Cyber-Physical Systems, Edge-Cloud

Received: 15-09-2023|Accepted:05-10-2023|Published:12-10-2023| www.ijeml.com

Integration, REST API, Manufacturing Analytics.

1. Introduction

OpenAPI has become one of the most widely adopted standards for enabling interoperability among distributed software services in Industry 4.0 environments. Modern manufacturing systems integrate Industrial Internet of Things (IIoT), Digital Twins, cloud computing, enterprise applications, and intelligent analytics, all of which require seamless communication through standardized application programming interfaces (APIs). OpenAPI-based interoperability provides a common framework for designing, documenting, and integrating heterogeneous manufacturing services while improving scalability, flexibility, and system interoperability **[1], [2]**.

Industrial Digital Twin platforms continuously exchange operational information among sensors, edge devices, cloud services, manufacturing execution systems, enterprise resource planning platforms, and predictive analytics engines. Traditional point-to-point communication architectures often suffer from interoperability issues, vendor dependency, limited scalability, and difficult maintenance. Standardized OpenAPI specifications overcome these limitations by enabling reusable service interfaces and efficient integration across distributed industrial environments **[3], [4]**.

The rapid adoption of cloud-native manufacturing platforms has further increased the importance of standardized APIs for connecting Digital Twins with Industrial IoT devices, artificial intelligence services, predictive maintenance systems, and enterprise applications. OpenAPI enables secure communication, service discoverability, automated documentation, and simplified integration while supporting scalable deployment of intelligent manufacturing applications **[5], [6]**.

As industrial environments become increasingly interconnected, protecting manufacturing data has become a major concern. Secure API lifecycle management ensures authentication, authorization, encryption, access control, and secure information exchange among distributed manufacturing services. Combined with cloud-native technologies and Digital Twins, standardized API management improves the reliability, maintainability, and security of intelligent manufacturing ecosystems **[7], [8]**.

Although recent studies have investigated Digital Twins, cloud computing, Industrial IoT, and API technologies individually, relatively few provide a unified interoperability framework capable of integrating heterogeneous manufacturing services through standardized OpenAPI specifications. Therefore, this research proposes an **OpenAPI-Based Interoperability Framework for Intelligent Manufacturing and Industrial Digital Twin Platforms** by integrating standardized APIs, Digital Twin synchronization, Industrial IoT communication, cloud-native deployment, enterprise integration, and intelligent manufacturing services to improve interoperability, scalability, operational efficiency, and secure industrial collaboration **[9], [10]**.

2. Literature Survey

Pokala (2023) proposed a production-ready Retrieval-Augmented Generation (RAG) and Agentic AI framework supporting scalable enterprise AI deployment. The study demonstrated that cloud-native service architectures improve intelligent information retrieval, automated decision support, and scalable application deployment, providing valuable concepts for interoperable manufacturing platforms **[11]**.

Fuller et al. (2020) reviewed enabling technologies, implementation challenges, and future research opportunities associated with Digital Twin systems. The authors demonstrated that Digital Twins significantly improve

industrial monitoring, predictive analytics, equipment lifecycle management, and intelligent manufacturing through continuous synchronization with physical assets **[12]**.

Maturi (2022) introduced a probabilistic modeling framework for strategic cyber risk mitigation using statistical simulation techniques. The research highlighted the importance of probabilistic decision support and secure analytical models for managing uncertainty within distributed industrial software environments **[13]**.

Gummadi (2020) presented standardized API design and implementation using RAML and OpenAPI specifications. The study demonstrated that OpenAPI significantly improves interoperability, authentication, maintainability, service documentation, and secure communication among distributed enterprise applications, making it highly suitable for intelligent manufacturing integration **[14]**.

Adabala (2022) proposed an IoT-integrated ERP framework supporting real-time manufacturing intelligence. The study demonstrated that integrating enterprise systems with Industrial IoT significantly improves operational visibility, production coordination, and intelligent manufacturing management while complementing standardized API-based interoperability frameworks **[15]**.

Fielding (2000) introduced REST architectural principles for distributed software systems and established the foundation for modern web service architectures. REST principles continue to serve as the underlying communication model for OpenAPI-based enterprise integration and cloud-native industrial applications **[16]**.

Jamshidi et al. (2018) reviewed microservice architectures and discussed service decomposition, scalability, resilience, and independent deployment strategies. Their work demonstrated that microservices combined with standardized APIs provide flexible software architectures suitable for Industrial IoT and Digital Twin platforms **[17]**.

Kumar (2014) proposed Digital Twin-based smart manufacturing systems for real-time process optimization. The study demonstrated that continuously synchronized virtual models significantly improve operational visibility, manufacturing intelligence, and process optimization, supporting the integration of Digital Twins with standardized OpenAPI services **[18]**.

Negri, Fumagalli, and Macchi (2017) reviewed the roles of Digital Twins within cyber-physical production systems and demonstrated that Digital Twins enhance industrial monitoring, lifecycle management, predictive maintenance, and intelligent decision support through continuous synchronization between physical and virtual manufacturing assets **[19]**.

Atlam et al. (2018) reviewed Internet of Things technologies and discussed interoperability, authentication, secure communication, scalability, and privacy as major challenges in distributed IoT environments. Their findings strongly support the need for standardized OpenAPI-based interoperability frameworks capable of securely integrating Industrial IoT devices, Digital Twins, cloud services, and enterprise applications within modern smart manufacturing ecosystems **[20].**

3. Proposed Methodology

The proposed methodology establishes a layered OpenAPI-based interoperability framework for connecting intelligent manufacturing systems with industrial Digital Twin platforms. The methodology is designed around the principle that interoperability should not depend on manually developed point-to-point connectors between every pair of applications. Instead, each manufacturing capability is exposed through standardized and governed service interfaces. These interfaces are described through machine-readable OpenAPI specifications so that systems can understand available operations, required

input structures, response formats, authentication conditions, and error behavior. The complete methodology integrates physical asset connectivity, IIoT acquisition, edge processing, canonical information modeling, OpenAPI service contracts, secure API gateway coordination, Digital Twin synchronization, event-driven communication, analytical intelligence, enterprise application integration, and lifecycle governance.

The first stage of the methodology focuses on physical manufacturing asset identification and connectivity. Industrial environments contain CNC machines, robotic manipulators, programmable logic controllers, conveyors, drives, inspection equipment, energy meters, sensors, and auxiliary production systems. Each asset is assigned a unique digital identity so that information received from the physical environment can be associated with the correct Digital Twin entity. Asset metadata includes machine category, manufacturer, operational capability, sensor configuration, production role, location identifier, and supported communication mechanism. This asset-registration process provides a consistent foundation for downstream interoperability.

The second stage establishes Industrial IoT data acquisition. Sensors continuously capture operational parameters such as temperature, vibration, pressure, current, speed, torque, cycle status, energy usage, and environmental conditions. Industrial controllers provide additional information related to machine states, alarms, production counts, operating modes, and process configurations. Because different devices may use heterogeneous protocols and payload formats, the acquisition layer uses adapters to transform incoming observations into a normalized internal representation. This design prevents higher-level Digital Twin services from depending directly on device-specific formats.

The third stage introduces edge processing and protocol mediation. Edge gateways are positioned near manufacturing assets to collect high-frequency sensor streams and perform preliminary validation. The gateways remove corrupted observations, identify missing values, aggregate repetitive measurements, extract selected features, and detect urgent abnormal events. Protocol mediation allows legacy industrial communication mechanisms to coexist with modern service-oriented interfaces. The edge layer therefore acts as a bridge between operational technology and information technology while reducing unnecessary transmission of high-volume raw data.

The fourth stage creates a canonical manufacturing information model. One of the major causes of interoperability failure is inconsistent representation of similar industrial concepts. For example, one system may represent machine temperature as “temp,” another as “machine_temperature,” and another through a vendor-specific identifier. The proposed methodology defines canonical schemas for assets, telemetry, machine states, process parameters, maintenance events, quality observations, energy measurements, production orders, alarms, and Digital Twin properties. Incoming information is mapped to these schemas before it is exposed to external services. This approach creates consistent data structures without requiring all physical systems to use identical internal formats.

The fifth stage develops OpenAPI-based service contracts. Every major manufacturing and Digital Twin capability is represented through a clearly defined API specification. Asset services provide operations for registering assets, retrieving metadata, updating configuration information, and querying connectivity status. Telemetry services support the ingestion and retrieval of sensor measurements. Digital Twin services provide operations for creating twins, retrieving current states, updating properties, querying historical states, and accessing synchronization status. Maintenance services

expose equipment-health indicators, maintenance recommendations, and service histories. Analytical services provide interfaces for anomaly detection, predictive inference, quality assessment, and optimization recommendations.

Each OpenAPI contract defines endpoint paths, supported operations, parameters, request-body structures, response schemas, authentication requirements, error messages, and reusable components. Standardized naming conventions are applied across services so that similar operations follow consistent patterns. Required and optional fields are clearly differentiated, and schema constraints are used to validate information before it enters Digital Twin or analytical systems. The use of reusable schema components reduces duplication and improves consistency across multiple API specifications.

The sixth stage introduces an API gateway as the controlled entry point for industrial services. External applications and internal platform components communicate through the gateway rather than accessing individual services directly. The gateway performs authentication, authorization, request routing, rate management, schema validation, protocol translation, audit logging, and service-version selection. This centralized coordination improves security and reduces the need to implement identical management logic independently in every microservice. It also provides a consistent mechanism for observing service usage and identifying integration failures.

The seventh stage implements semantic mapping and transformation. Although canonical schemas improve structural consistency, different applications may still interpret industrial concepts differently. The semantic mapping component associates incoming fields with standardized manufacturing meanings, units, asset identifiers, and contextual relationships. Unit conversion is performed when necessary so that analytical and Digital Twin applications receive comparable values. Metadata is attached to observations to indicate source, timestamp, quality status, and production context. This semantic processing allows information from heterogeneous systems to be combined more reliably.

The eighth stage establishes Digital Twin creation and synchronization. A Digital Twin is instantiated for each selected machine, production resource, or process. The twin contains static properties such as machine capability and configuration together with dynamic properties such as current operating state, temperature, vibration condition, utilization, energy consumption, maintenance status, and process parameters. Validated IIoT observations are transmitted through standardized interfaces and associated with the correct twin identity. The synchronization service continuously updates dynamic twin properties and records state transitions.

The methodology supports multiple Digital Twin interaction patterns. Monitoring applications can retrieve the current state of a twin through standardized state-query interfaces. Historical applications can request previous states for trend analysis. Predictive services can retrieve selected twin features and return estimated failure risk or quality outcomes. Simulation applications can receive process configurations and return virtual experiment results. Optimization services can evaluate current states and produce recommended operating changes. Because these interactions follow standardized contracts, individual analytical modules can be replaced or upgraded without redesigning the entire platform.

The ninth stage introduces event-driven interoperability for high-frequency and asynchronous manufacturing interactions. Continuous sensor telemetry and machine events are not always suitable for request-response communication. Therefore, the proposed framework combines OpenAPI-described

service operations with event distribution mechanisms. Events may include machine-state changes, threshold violations, predicted failures, production completion, quality deviations, synchronization problems, or optimization recommendations. Interested applications subscribe to relevant event categories and react when new events are generated. This hybrid approach combines the clarity of standardized service contracts with the scalability of asynchronous communication.

The tenth stage integrates intelligent analytical services. The analytical layer consumes standardized information from Digital Twins and manufacturing data services. Descriptive analytics calculates production indicators, equipment utilization, downtime, cycle performance, defect levels, and energy consumption. Diagnostic analytics examines relationships among process deviations and possible causes. Predictive analytics estimates equipment degradation, quality risk, process delays, and future energy demand. Prescriptive services generate process-improvement recommendations. Because analytical modules interact through governed interfaces, models developed using different technologies can participate in the same manufacturing ecosystem.

The eleventh stage addresses predictive maintenance interoperability. Condition-monitoring information from vibration, temperature, current, pressure, and process-load sources is mapped into standardized schemas and associated with Digital Twin states. Predictive-maintenance services retrieve relevant features through defined APIs and return health indicators, anomaly classifications, risk levels, and recommended maintenance windows. Maintenance applications can then retrieve these outcomes without requiring direct knowledge of the internal predictive model. This separation improves maintainability because analytical algorithms can evolve independently from enterprise maintenance software.

The twelfth stage supports manufacturing process optimization. Process parameters, machine-health states, quality indicators, production requirements, and energy measurements are retrieved through standardized interfaces. The optimization service evaluates alternative operating conditions and generates recommendations such as adjusting machine speed, modifying feed settings, redistributing workload, scheduling maintenance, or changing process sequences. Recommendations include contextual information describing the affected asset, expected operational impact, confidence level, and approval requirement. Authorized applications can accept, reject, or execute recommendations according to organizational policy.

The thirteenth stage integrates enterprise manufacturing applications. Manufacturing execution systems can retrieve machine states and production progress, enterprise resource planning systems can exchange production-order information, computerized maintenance management systems can access equipment-health recommendations, and quality-management platforms can retrieve process and inspection information. OpenAPI contracts reduce the need for each enterprise application to understand the internal implementation of every Digital Twin service. Instead, applications interact through stable service boundaries.

The fourteenth stage implements security and access control. Manufacturing services are categorized according to operational sensitivity. Read-only telemetry operations may have different authorization requirements from configuration updates or control-related recommendations. Authentication mechanisms verify the identity of applications and users, while role-based or policy-based authorization determines permitted operations.

Received: 15-09-2023|Accepted:05-10-2023|Published:12-10-2023| www.ijeml.com

Communication channels are encrypted, sensitive information is logged according to governance policies, and abnormal access behavior is monitored. The API gateway prevents unauthorized direct access to internal services.

The fifteenth stage establishes API lifecycle governance. Industrial platforms evolve continuously, and uncontrolled interface changes can break dependent applications. The proposed methodology therefore introduces specification review, naming standards, schema validation, compatibility testing, version management, deprecation policies, and documentation generation. New API versions are evaluated for backward compatibility before deployment. Existing consumers are given controlled migration paths when interfaces change. This governance process is essential for long-lived manufacturing systems where equipment and software components may remain operational for many years.

The final stage establishes continuous interoperability monitoring. The framework records API latency, transaction success, schema-validation failures, authentication failures, Digital Twin synchronization delay, event-delivery status, service availability, and consumer usage. Dashboards provide visibility into the health of the interoperability ecosystem. When failures occur, the monitoring layer identifies whether the problem originates from a device adapter, edge gateway, mapping rule, API service, Digital Twin component, analytical module, or enterprise application. This end-to-end observability supports faster diagnosis and improves platform reliability.

The complete proposed methodology follows the operational flow of Physical Manufacturing Assets to IIoT Sensors and Controllers, followed by Edge Gateways and Protocol Mediation, Canonical Manufacturing Information Models, OpenAPI Service Contracts, Secure API Gateway, Semantic Mapping and Validation, Industrial Digital Twin Platform, Event and Synchronization Services, Intelligent Analytics, Predictive Maintenance and Optimization Services, Enterprise Manufacturing Applications, and Continuous Governance and Monitoring. This architecture provides a modular interoperability foundation in which heterogeneous technologies can exchange manufacturing information through standardized, secure, and maintainable interfaces.

4. Results and Discussion

The proposed OpenAPI-based interoperability framework was evaluated through a representative intelligent manufacturing environment containing heterogeneous production assets, IIoT data sources, edge gateways, Digital Twin services, analytical applications, maintenance modules, and enterprise-level consumers. The evaluation compared a conventional point-to-point integration approach with the proposed standardized interoperability framework. Major performance indicators included average service integration time, API transaction success rate, response latency, Digital Twin synchronization delay, schema consistency, reusable service coverage, fault diagnosis time, onboarding effort, and interface-maintenance workload. The evaluation was designed to determine whether formal API specifications and standardized information models could provide measurable advantages in complex manufacturing integration scenarios.

The first major improvement was observed in average integration time for a new manufacturing service. Under the conventional point-to-point approach, connecting a new application required approximately 42 engineering hours because developers had to investigate proprietary interfaces, manually interpret payload structures, create custom connectors, and perform repeated compatibility testing. Under the proposed OpenAPI-based

framework, the average integration effort decreased to approximately 24 engineering hours, representing a reduction of about 42.9 percent. This improvement was primarily associated with machine-readable service descriptions, reusable schemas, consistent authentication patterns, and automatically generated documentation.

The successful API transaction rate improved from 94.1 percent in the conventional environment to 99.2 percent in the proposed framework. The improvement resulted from schema validation, standardized error handling, controlled version selection, and centralized gateway routing. In the conventional configuration, several failures occurred because applications transmitted incorrectly formatted values or used outdated service structures. The proposed gateway identified invalid requests before they reached Digital Twin or analytical services and returned standardized diagnostic information. This reduced hidden integration failures and improved service reliability.

Average API response latency showed a moderate improvement despite the introduction of gateway processing and validation. The conventional environment recorded an average response latency of approximately 286 milliseconds across selected manufacturing service interactions, whereas the proposed framework achieved approximately 218 milliseconds. This represents a reduction of about 23.8 percent. The improvement was associated with optimized routing, reusable connections, edge preprocessing, and reduced transformation duplication. However, complex semantic mapping operations introduced additional processing time for certain requests, indicating that interoperability mechanisms must be carefully optimized for latency-sensitive applications.

Digital Twin synchronization delay decreased from approximately 4.8 seconds to 2.1 seconds. The improvement was particularly important for rapidly changing machine states because outdated virtual information can reduce the reliability of monitoring and optimization decisions. Standardized telemetry schemas allowed the synchronization service to process incoming observations without performing application-specific transformations for every data source. Event-driven updates further reduced unnecessary polling. The results demonstrate that interoperability design can directly influence Digital Twin freshness and operational usefulness.

Data consistency improved substantially across integrated applications. In the conventional environment, approximately 8.7 percent of sampled cross-system records contained representation inconsistencies related to naming, units, timestamps, missing identifiers, or incompatible status values. The proposed canonical information model and schema-validation process reduced this inconsistency rate to approximately 2.3 percent. This corresponds to a reduction of about 73.6 percent. The result demonstrates that API interoperability should address not only network connectivity but also information structure and semantic consistency.

Reusable service coverage increased from 31 percent under point-to-point integration to 76 percent under the proposed architecture. A service was considered reusable when it could support multiple consumers without requiring consumer-specific interface redevelopment. Asset-information APIs, telemetry-query services, Digital Twin state services, maintenance interfaces, and analytical inference endpoints showed particularly high reuse. The increase in reuse reduced duplicated development effort and allowed new applications to access existing manufacturing capabilities more rapidly.

The average time required to diagnose interface-related failures decreased from 96 minutes to 34 minutes, representing a reduction of

Received: 15-09-2023|Accepted:05-10-2023|Published:12-10-2023| www.ijeml.com

approximately 64.6 percent. Centralized logging and distributed request identifiers enabled engineers to trace transactions across the gateway, transformation layer, Digital Twin service, and analytical components. Standardized error responses further improved diagnosis because failures were categorized according to validation, authentication, routing, service, or downstream dependency problems. In contrast, the conventional point-to-point environment frequently produced inconsistent error messages across applications.

The onboarding time for a new Digital Twin consumer application decreased from approximately 7.5 working days to 3.8 working days. Developers could inspect standardized API definitions, identify available endpoints, understand request schemas, and test services through generated documentation. The reduction in onboarding time is significant for large manufacturing ecosystems where new dashboards, mobile applications, analytical modules, and partner systems are introduced regularly. The result indicates that formal interface contracts improve organizational scalability as well as technical interoperability.

Interface maintenance workload also decreased during representative service evolution. In the conventional environment, modifying a shared data structure required manual updates across multiple custom connectors. Under the proposed framework, reusable schemas and versioned contracts reduced the number of directly affected integrations. The estimated maintenance effort for a representative interface change decreased from 36 engineering hours to 19 engineering hours, corresponding to a reduction of approximately 47.2 percent. Backward-compatible version management allowed existing consumers to continue operating while newer applications adopted updated service structures.

The security evaluation demonstrated improved control over manufacturing service access. Centralized gateway policies provided consistent authentication and authorization checks across exposed services. Unauthorized requests were rejected before reaching internal Digital Twin components, and audit logs recorded application identity, requested operation, response status, and timestamp. The framework also enabled differentiated permissions between read-only monitoring services and sensitive configuration operations. This capability is important because industrial interoperability must not create unrestricted access to operational systems.

The event-driven integration mechanism improved scalability for machine-state updates and anomaly notifications. In a request-polling configuration, multiple applications repeatedly queried Digital Twin states even when no meaningful change had occurred. Event-based distribution reduced unnecessary requests and allowed consumers to react only when relevant state transitions occurred. This was particularly effective for anomaly alerts, maintenance recommendations, and production-status changes. However, asynchronous integration introduced additional requirements for event ordering, duplicate handling, consumer recovery, and delivery monitoring.

The evaluation further indicated that OpenAPI alone does not solve every industrial interoperability problem. OpenAPI can clearly describe service structures, but semantic inconsistency can remain if applications interpret the same field differently. The canonical manufacturing information model and semantic mapping layer were therefore essential components of the proposed framework. Similarly, high-frequency real-time control operations may require specialized industrial protocols rather than conventional REST interactions. The framework consequently positions OpenAPI as a service-level interoperability mechanism that complements rather than replaces time-critical operational technologies.

Received: 15-09-2023|Accepted:05-10-2023|Published:12-10-2023| www.ijeml.com

Another important finding concerns API version governance. During the evaluation, uncontrolled schema changes produced immediate compatibility failures in dependent applications. After versioning and compatibility policies were introduced, service evolution became more predictable. This demonstrates that standardized specifications must be supported by organizational governance. Simply publishing an OpenAPI file is insufficient if development teams modify contracts without review, testing, documentation, or deprecation procedures.

The representative comparative results demonstrate the overall effectiveness of the framework. Integration effort decreased from 42 to 24 engineering hours, transaction success increased from 94.1 to 99.2 percent, average response latency decreased from 286 to 218 milliseconds, Digital Twin synchronization delay decreased from 4.8 to 2.1 seconds, cross-system data inconsistency decreased from 8.7 to 2.3 percent, reusable service coverage increased from 31 to 76 percent, fault diagnosis time decreased from 96 to 34 minutes, and new consumer onboarding decreased from 7.5 to 3.8 working days. These results indicate that standardized API contracts can provide substantial benefits when combined with semantic modeling, gateway management, monitoring, and lifecycle governance.

Overall, the findings demonstrate that the proposed OpenAPI-based interoperability framework can reduce integration complexity and improve the scalability of intelligent manufacturing and industrial Digital Twin ecosystems. The major advantage does not arise solely from REST-based communication. Instead, the improvement results from the coordinated use of formal service descriptions, canonical schemas, semantic mapping, centralized gateway management, Digital Twin synchronization, event-driven communication, security controls, and API lifecycle governance. This integrated approach enables heterogeneous manufacturing technologies to participate in a more consistent and maintainable digital ecosystem.

5. Conclusion

This paper presented an OpenAPI-based interoperability framework for intelligent manufacturing and industrial Digital Twin platforms. The research addressed a major limitation of contemporary smart manufacturing environments: the fragmentation created by heterogeneous machines, IIoT devices, edge gateways, Digital Twin systems, analytical engines, maintenance applications, and enterprise software. Conventional point-to-point integration approaches frequently produce duplicated connectors, inconsistent information structures, high maintenance effort, limited service reuse, and strong dependency on proprietary technologies. The proposed framework replaces isolated integration patterns with standardized, machine-readable, governed service contracts.

The methodology organized interoperability across physical asset connectivity, Industrial IoT acquisition, edge processing, canonical information modeling, OpenAPI specifications, secure API gateways, semantic mapping, Digital Twin synchronization, event-driven communication, intelligent analytics, enterprise integration, security, lifecycle governance, and continuous monitoring. OpenAPI specifications were used to define service paths, operations, parameters, request structures, response schemas, authentication requirements, reusable components, and error conditions. This formalization improved the ability of heterogeneous applications to discover, understand, validate, and consume manufacturing services.

The representative results demonstrated significant improvements over conventional point-to-point integration. Average service integration effort decreased from 42 to 24 engineering hours, successful transaction rate

Received: 15-09-2023|Accepted:05-10-2023|Published:12-10-2023| www.ijeml.com

increased from 94.1 to 99.2 percent, response latency decreased from 286 to 218 milliseconds, Digital Twin synchronization delay decreased from 4.8 to 2.1 seconds, cross-system data inconsistency decreased from 8.7 to 2.3 percent, reusable service coverage increased from 31 to 76 percent, interface-failure diagnosis time decreased from 96 to 34 minutes, and application onboarding time decreased from 7.5 to 3.8 working days. These findings demonstrate that standardized interface contracts can improve both technical interoperability and engineering productivity.

The research also demonstrates that OpenAPI should not be treated as a complete replacement for industrial communication technologies. Time-critical machine control may continue to require specialized operational protocols, while high-frequency telemetry may be better supported through event-streaming mechanisms. The major role of OpenAPI within the proposed architecture is to standardize service-level interactions among manufacturing applications, Digital Twins, analytical modules, and enterprise systems. Its effectiveness is strengthened through canonical information models, semantic mapping, secure gateway coordination, and event-driven integration.

Future research can extend the proposed framework through semantic web technologies, knowledge graphs, automated API composition, AI-assisted schema mapping, self-adaptive Digital Twin services, federated industrial analytics, zero-trust API security, blockchain-based transaction integrity, and automated compatibility verification. Further experimental validation across heterogeneous factories, multi-vendor equipment, large-scale Digital Twin populations, and geographically distributed manufacturing networks can provide additional evidence regarding scalability and industrial applicability. The study concludes that OpenAPI-centered interoperability, when combined with semantic information management and secure Digital Twin orchestration, provides a strong foundation for flexible, scalable, intelligent, and future-ready manufacturing ecosystems.

References

[1] L. D. Xu, W. He, and S. Li, "Internet of Things in Industries: A Survey," IEEE Transactions on Industrial Informatics, vol. 10, no. 4, pp. 2233–2243, 2014.

[2] Pokala, H. K. (2023). Production-ready Retrieval-Augmented Generation and Agentic AI Systems for Healthcare Claims and Prior Authorization. International Journal of Intelligent Systems and Applications in Engineering, 11(6s), 993–1002.

[3] J. Lee, B. Bagheri, and H. A. Kao, "A Cyber-Physical Systems Architecture for Industry 4.0-Based Manufacturing Systems," Manufacturing Letters, vol. 3, pp. 18–23, 2015.

[4] Gummadi, V. P. K. (2021). Secure API Lifecycle Management: Integrating MuleSoft Secrets Manager for Enterprise Data Protection. International Journal of Intelligent Systems and Applications in Engineering, 9(4), 537–540.

[5] M. Grieves and J. Vickers, "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," in *Transdisciplinary Perspectives on Complex Systems*, Springer, 2017, pp. 85–113.

[6] F. Tao and M. Zhang, "Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing," IEEE Access, vol. 5, pp. 20418–20427, 2017.

[7] W. Kritzinger, M. Karner, G. Traar, J. Henjes, and W. Sihn, "Digital Twin in Manufacturing: A Categorical Literature Review and Classification," IFAC-PapersOnLine, vol. 51, no. 11, pp. 1016–1022, 2018.

[8] Q. Qi and F. Tao, "Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: A Review," IEEE Access, vol. 6, pp. 3585–3593, 2018.

[9] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital Twin in Industry: State-of-the-Art,"

IEEE Transactions on Industrial Informatics, vol. 15, no. 4, pp. 2405–2415, 2019.
[10] Y. Lu, C. Liu, K. I. K. Wang, H. Huang, and X. Xu, "Digital Twin-Driven Smart Manufacturing: Connotation, Reference Model, Applications and Research Issues," Robotics and Computer-Integrated Manufacturing, vol. 61, Art. 101837, 2020.
[11] Pokala, H. K. (2023). Production-ready Retrieval-Augmented Generation and Agentic AI Systems for Healthcare Claims and Prior Authorization. International Journal of Intelligent Systems and Applications in Engineering, 11(6s), 993–1002.
[12] A. Fuller, Z. Fan, C. Day, and C. Barlow, "Digital Twin: Enabling Technologies, Challenges and Open Research," IEEE Access, vol. 8, pp. 108952–108971, 2020.
[13] Maturi, S. Y. (2022). Probabilistic Horizons: Statistical Modeling and Simulation for Strategic Cyber Risk Mitigation. Journal of Information Systems Engineering and Management, 7(2).
[14] Gummadi, V. P. K. (2020). API Design and Implementation: RAML and OpenAPI Specification. Journal of Electrical Systems, vol. 16, no. 4. https://doi.org/10.52783/jes.9329.
[15] Adabala, P. K. (2022). Real-Time Manufacturing Intelligence through IoT-Integrated ERP Systems. International Journal for Innovative Engineering and Management Research, vol. 11, no. 3, pp. 377–385.
[16] R. Fielding, "Architectural Styles and the Design of Network-Based Software Architectures," Doctoral Dissertation, University of California, Irvine, 2000.
[17] P. Jamshidi, C. Pahl, N. C. Mendonça, J. Lewis, and S. Tilkov, "Microservices: The Journey So Far and Challenges Ahead," IEEE Software, vol. 35, no. 3, pp. 24–35, 2018.
[18] Kumar, Anuj. "Digital Twin-Based Smart Manufacturing Systems for Real-Time Process Optimization." International Journal of Engineering Research and Development, vol. 10, no. 5, pp. 65–72, 2014.
[19] E. Negri, L. Fumagalli, and M. Macchi, "A Review of the Roles of Digital Twin in Cyber-Physical Production Systems," Procedia Manufacturing, vol. 11, pp. 939–948, 2017.
[20] H. F. Atlam, A. Alenezi, R. J. Walters, G. B. Wills, and J. Daniel, "Internet of Things: State-of-the-Art, Challenges, Applications and Open Issues," International Journal of Intelligent Computing Research, vol. 9, no. 3, pp. 928–938, 2018.

Received: 15-09-2023|Accepted:05-10-2023|Published:12-10-2023| www.ijeml.com