World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Check for updates

Integrating Jenkins and AWS in Siemens Opcenter MES: Bridging the digital divide through modern development and deployment pipelines

Satish Kumar Nalluri * and Varun Teja Bathini

[1] *Sr Developer, Life Sciences, Thermo Fisher Scientific, Inc., Frederick, Maryland, USA.*
[2] *MES Functional Analyst III, Life Sciences, Penumbra, Inc., Alameda, California, USA.*

World Journal of Advanced Engineering Technology and Sciences, 2025, 15(02), 2064-2074

Publication history: Received on 05 April 2025; revised on 14 May 2025; accepted on 17 May 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.15.2.0767

Abstract

The integration of Jenkins and Amazon Web Services (AWS) into Siemens Opcenter MES (Manufacturing Execution System) represents a transformative approach to modernizing industrial software deployment. As manufacturing systems evolve toward Industry 4.0, the need for agile, scalable, and automated deployment pipelines becomes critical. This paper explores how Continuous Integration and Continuous Deployment (CI/CD) methodologies, facilitated by Jenkins automation and AWS cloud infrastructure, can bridge the digital divide in traditional MES environments.

We present a structured framework for automating build, test, and deployment processes in Siemens Opcenter, leveraging AWS services (EC2, ECS, Lambda, CloudFormation) for scalable and secure execution. Key benefits include reduced deployment times, enhanced reliability, and cost-efficient scalability, while addressing challenges such as security, compliance, and system dependencies. A real-world case study demonstrates measurable improvements in deployment speed and operational efficiency, validating the proposed approach.

Finally, we discuss emerging trends, including AI-driven deployment optimization, serverless architectures, and edge computing, highlighting future directions for MES DevOps. This research provides actionable insights for manufacturing IT teams seeking to adopt modern CI/CD pipelines while maintaining robustness in industrial environments.

Keywords: Siemens Opcenter MES; Jenkins CI/CD; AWS Cloud Integration; Manufacturing Execution System (MES); Industry 4.0; DevOps Automation; Continuous Deployment Pipelines; Infrastructure as Code (IaC); Cloud-Based Manufacturing; Scalable Industrial DevOps

1. Introduction

The manufacturing industry is undergoing a digital transformation driven by Industry 4.0 technologies, requiring Manufacturing Execution Systems (MES) to become more agile, scalable, and automated. Siemens Opcenter MES plays a crucial role in this transformation by bridging enterprise planning systems with shop-floor operations, enabling real-time production monitoring, quality control, and data-driven decision making (Siemens, 2024). However, traditional MES deployments often rely on manual processes that lead to slow update cycles, deployment errors, and limited scalability - challenges that hinder manufacturers from fully realizing Industry 4.0 benefits.

Continuous Integration and Continuous Deployment (CI/CD) pipelines have emerged as critical enablers for modern MES implementations. By automating software build, testing, and deployment processes, CI/CD significantly reduces time-to-market while improving system reliability (Sharma et al., 2023). Jenkins, as an open-source automation server, provides the foundation for establishing these pipelines, while Amazon Web Services (AWS) offers the cloud

* Corresponding author: Nalluri Satish Kumar

infrastructure needed for scalable, secure deployments. Together, they form a powerful combination that can transform how manufacturers deploy and update their MES solutions (AWS, 2024).

This paper explores the integration of Jenkins and AWS with Siemens Opcenter MES to create modern development and deployment pipelines. The research focuses on three key objectives:

- accelerating deployment speed through automation,
- improving system reliability through automated testing and rollback mechanisms, and
- Enabling elastic scalability to handle fluctuating production demands.

By examining architectural approaches, implementation strategies, and real-world applications, this study provides valuable insights for manufacturing organizations seeking to modernize their MES infrastructure.

The significance of this integration extends beyond technical implementation. It represents a fundamental shift in how manufacturing software is developed and deployed, enabling faster innovation cycles and more responsive production systems. As noted by Müller et al. (2022), the ability to rapidly deploy MES updates is becoming a competitive differentiator in industries ranging from automotive to pharmaceuticals. This paper contributes to both academic research and industrial practice by presenting a comprehensive framework for implementing CI/CD in MES environments, supported by case evidence and technical validation.

2. Understanding the key components

2.1. Siemens Opcenter MES

In the age of Industry 4.0, manufacturing companies around the world are shifting from conventional systems to digital, automated platforms that can enhance visibility, traceability, and efficiency. One such powerful platform is Siemens Opcenter Manufacturing Execution System (MES). This system acts as a bridge between enterprise-level software, such as Enterprise Resource Planning (ERP) systems, and plant-floor operations, helping manufacturers gain real-time insights and control over their production processes.

2.1.1. Core Functionalities and Architecture

Siemens Opcenter MES provides a wide array of functionalities that help manufacturers streamline production operations. These core functionalities are designed to cover every aspect of the production process, from planning to execution and analysis. Let's explore these capabilities in detail:

- **Production Scheduling and Dispatching:** One of the central roles of Opcenter MES is to manage and schedule production orders. It enables real-time scheduling by considering constraints like resource availability, shift calendars, and machine downtime. This leads to optimized production flow and better utilization of manufacturing resources.
- **Real-Time Data Collection and Monitoring:** Opcenter MES collects and analyzes data from machines and operators in real time. Sensors and devices connected through Industrial Internet of Things (IIoT) send production and machine status updates, which can be monitored live. This ensures transparency and quick decision-making (Siemens, 2023).
- **Quality Management:** The MES system supports integrated quality management by monitoring production parameters, performing statistical process control (SPC), and initiating corrective actions when needed. This reduces waste and ensures that products meet quality standards throughout the production lifecycle.
- **Traceability and Genealogy:** One of the vital features of Opcenter MES is complete traceability. It records every component, material, and operation involved in producing a product. This is particularly important in industries like pharmaceuticals, food, and aerospace, where strict compliance and recall capabilities are necessary.
- **Performance Analysis and KPIs:** Opcenter provides tools for tracking Key Performance Indicators (KPIs) such as Overall Equipment Effectiveness (OEE), production yield, and downtime. These metrics help managers identify bottlenecks, inefficiencies, and areas for improvement.
- **Integration with Other Systems:** Opcenter MES is designed with open architecture to integrate seamlessly with ERP systems (like SAP), product lifecycle management (PLM) software, and automation systems such as SCADA and DCS. This integration creates a unified manufacturing ecosystem where data flows freely and processes are synchronized (Kumar et al., 2021).

- **System Architecture Overview:** Opcenter MES is built on a scalable, modular architecture. The system consists of multiple modules that serve specific functions such as planning, execution, quality, logistics, and performance. It is typically deployed using a client-server model, where a central server hosts the MES software and clients access the system via web or desktop interfaces.

The architecture is also cloud-compatible, meaning organizations can deploy it on-premises, in a hybrid cloud environment, or fully on the cloud. Siemens offers Opcenter with support for Microsoft Azure and other major cloud platforms. This flexibility ensures that companies of all sizes and IT maturity levels can adopt it.

In addition, Opcenter supports microservices-based architecture and APIs (Application Programming Interfaces) that allow for the extension and customization of its features. This modern design enables organizations to adapt the MES to their specific needs without being locked into a rigid structure.

2.1.2. Challenges in Traditional Deployment

While Siemens Opcenter MES offers a powerful suite of tools, traditional deployment methods come with a variety of challenges. These issues often hinder the rapid implementation and scalability that modern manufacturing environments demand.

- **Long Deployment Cycles:** Traditional MES deployment typically involves extensive customization, on-premise setup, and integration with legacy systems. These activities can take months or even years to complete. The complexity of mapping MES functions to the unique workflows of each factory adds to the delay. As a result, businesses may experience disruptions in operations during the transition period.
- **High Initial Costs:** Deploying a traditional MES like Opcenter on-premises usually requires a significant capital investment. This includes the cost of servers, networking infrastructure, software licenses, training, and consultant fees. For small- and medium-sized enterprises (SMEs), these costs can be prohibitive.
- **Integration Complexity:** Although Opcenter is designed to integrate with ERP, PLM, and automation systems, achieving seamless integration in traditional deployment environments can be difficult. Legacy systems often lack standardized communication protocols, and many production lines operate on outdated PLCs or SCADA systems that are not compatible out-of-the-box with modern MES platforms (Lee et al., 2020).
- **Limited Scalability and Flexibility:** In a traditional setup, scaling the system to new plants or production lines involves re-implementation or significant configuration changes. This makes it difficult for global manufacturers to maintain a standardized MES framework across locations. The lack of flexibility can also hinder the adoption of new technologies like AI, machine learning, and cloud computing.
- **Maintenance and Downtime:** On-premises MES deployments require ongoing maintenance for hardware and software. Updates, backups, and system monitoring need to be performed regularly to ensure reliability. Downtime for system upgrades or unexpected failures can disrupt production schedules and lead to financial losses.
- **Resistance to Change:** Cultural and organizational resistance is another challenge. Factory workers, engineers, and supervisors who are used to manual processes may be hesitant to adopt a digital system. Overcoming this resistance requires comprehensive training programs and change management initiatives.
- **Data Silos and Inconsistent Standards:** In traditional deployments, data is often stored in silos across different systems, making it difficult to extract holistic insights. Moreover, inconsistent data standards and naming conventions across departments can reduce the effectiveness of MES analytics.
- **Cybersecurity Concerns:** On-premise systems may lack the robust cybersecurity protections that cloud-based systems can offer. This exposes manufacturing plants to risks like data breaches, ransomware attacks, and intellectual property theft. As digital transformation progresses, security becomes a more pressing concern (Chen et al., 2021).

2.2. Jenkins for CI/CD

In modern software development and system integration, automation is a key enabler of efficiency and consistency. Continuous Integration and Continuous Deployment (CI/CD) pipelines are central to DevOps practices, ensuring that changes in code are tested, validated, and deployed automatically. One of the most widely used tools for CI/CD is **Jenkins**, an open-source automation server that has become a standard in software development and operations.

2.2.1. Automation of Builds, Tests, and Deployments

Jenkins plays a critical role in automating software pipelines, especially for systems that require frequent updates and testing—like **Manufacturing Execution Systems (MES)**. When developers update or add new code to a shared repository (e.g., Git), Jenkins can be triggered automatically to start a build process.

- **Build Automation:** The build process involves compiling source code into executable files, generating configuration files, and packaging the software for deployment. Jenkins supports many build tools such as Maven, Gradle, and Ant. Once set up, this process requires no manual intervention, reducing human errors and accelerating the development cycle.
- **Automated Testing:** After the build, Jenkins can automatically execute a suite of tests—unit tests, integration tests, and even user interface (UI) tests. Automated testing ensures that new code does not break existing functionality and maintains the quality of the software. This is especially important in MES environments, where even a small bug could halt manufacturing operations.
- **Deployment Automation:** Following successful builds and tests, Jenkins can deploy the application to a staging or production environment. This deployment may include restarting servers, updating configurations, or copying files to cloud storage. For example, in the case of deploying Siemens Opcenter MES extensions or microservices, Jenkins can push updates to containerized environments using Docker or Kubernetes (Gupta & Sharma, 2021).
- **Continuous Feedback and Monitoring:** Jenkins provides real-time feedback to developers. If a build or test fails, developers are notified instantly, allowing them to fix issues quickly. Dashboards and plugins like Jenkins Blue Ocean make pipeline visualizations easier, providing transparency and accountability in development teams.

2.2.2. Plugins and Extensibility for MES Environments

One of Jenkins' greatest strengths is its **extensibility**. Jenkins has over 1,800 community-contributed plugins, which allow it to integrate with almost every tool and platform used in modern DevOps environments.

- **Integration with MES Platforms:** In MES environments, Jenkins can integrate with manufacturing software through APIs and RESTful services. For example, it can call scripts that deploy updates to Siemens Opcenter's middleware or data services, or send deployment logs to a centralized monitoring tool.
- **SCM and Artifact Repositories:** Jenkins integrates with Source Code Management (SCM) systems like GitHub, GitLab, and Bitbucket. It also works with artifact repositories like Nexus and JFrog Artifactory, which are used to manage compiled MES modules, libraries, and configuration files.
- **Infrastructure as Code (IaC):** With plugins such as Terraform, Ansible, and AWS CloudFormation, Jenkins can also provision infrastructure automatically. This is useful when setting up environments for MES applications, especially in cloud deployments where infrastructure may be ephemeral.
- **Docker and Kubernetes:** For MES environments moving toward containerization, Jenkins provides excellent support for Docker and Kubernetes. Developers can package MES components in Docker containers and deploy them in Kubernetes clusters with Jenkins pipelines. This approach reduces environment-specific issues and increases portability (Rahman et al., 2020).

2.3. AWS Cloud Services

Cloud computing platforms, particularly Amazon Web Services (AWS), offer a robust foundation for deploying and managing MES systems like Siemens Opcenter. AWS provides on-demand scalability, security, and a suite of tools tailored for enterprise-grade applications.

2.3.1. Key Services for Deployment (EC2, ECS, Lambda, S3, Code Deploy)

AWS offers a rich ecosystem of services that are particularly useful for deploying and maintaining modern MES and CI/CD pipelines.

- **Amazon EC2 (Elastic Compute Cloud):** EC2 provides resizable virtual servers that can run MES applications, databases, and supporting services. These virtual machines can be scaled up or down based on demand, allowing for efficient resource management.
- **Amazon ECS (Elastic Container Service):** ECS is AWS's container orchestration platform. It allows developers to run Docker containers across a fleet of EC2 instances. This is ideal for containerized MES components, where tasks such as scheduling, load balancing, and service discovery are managed automatically by ECS.

- **AWS Lambda:** Lambda enables **serverless computing**, where developers can run code in response to events without managing servers. In MES environments, Lambda can be used for lightweight tasks like triggering updates, processing logs, or performing scheduled data cleanup. Since Lambda is charged based on execution time, it is a cost-effective solution for auxiliary MES functions.
- **Amazon S3 (Simple Storage Service):** S3 provides scalable object storage and is widely used for storing software artifacts, log files, configuration files, and MES-generated reports. S3 integrates with other AWS services for secure data sharing and analytics.
- **AWS Code Deploy:** Code Deploy automates application deployments to EC2, ECS, Lambda, or on-premises servers. It ensures that deployments are predictable, consistent, and can be rolled back if needed. This is critical for MES platforms where high availability and minimum downtime are essential.

By combining these services, organizations can build a fully automated CI/CD pipeline and deploy MES components efficiently and securely in the cloud.

2.3.2. Scalability and Security Considerations

- **Scalability:** AWS provides vertical and horizontal scalability. Vertical scaling involves increasing the resources (CPU, RAM) of EC2 instances. Horizontal scaling, often managed through AWS Auto Scaling, allows the system to increase or decrease the number of instances based on demand. This ensures that MES applications can handle load fluctuations, such as increased production shifts or seasonal demands.

Container orchestration with ECS or Kubernetes on AWS (via EKS) also supports auto-scaling based on metrics such as CPU usage or queue size, making it easier to manage resource consumption efficiently.

- **Security:** Security is paramount in manufacturing environments. AWS provides multiple layers of security controls to protect MES deployments:
    - **Identity and Access Management (IAM):** Controls who can access what resources.
    - **Encryption:** AWS offers encryption at rest (e.g., using KMS for EC2 volumes or S3 buckets) and in transit (SSL/TLS).
    - **VPC and Networking Controls:** Virtual Private Cloud (VPC) enables the creation of isolated network environments with controlled access points.
    - **Logging and Auditing:** AWS CloudTrail and Amazon CloudWatch enable activity monitoring, alerting, and auditing for compliance.

Moreover, AWS complies with industry standards such as ISO 27001, SOC 2, and NIST, making it suitable for regulated industries including pharmaceuticals, aerospace, and automotive manufacturing.

The integration of Jenkins CI/CD pipelines with Siemens Opcenter MES and the deployment of these systems on AWS Cloud Services represents a powerful combination for modern smart manufacturing. Jenkins automates critical tasks—builds, testing, and deployment—ensuring rapid and error-free software delivery. Its extensibility allows seamless integration into MES environments, while AWS offers the scalability, reliability, and security required to run these complex systems in production.

The use of AWS services like EC2, ECS, Lambda, and S3 provides a scalable infrastructure backbone, and tools like Code Deploy streamline software releases. Furthermore, the cloud-native approach enhances flexibility and supports global operations by reducing dependency on on-premises infrastructure.

When combined, Jenkins and AWS provide a robust foundation for deploying Siemens Opcenter MES in a modern, cloud-integrated environment. This synergy supports the goals of Industry 4.0 by enhancing agility, reducing downtime, and improving operational excellence.

3. Bridging the gap: integration strategy

Integrating Siemens Opcenter MES with Jenkins CI/CD pipelines and AWS cloud infrastructure requires a well-planned strategy to ensure consistency, security, and performance. This section explores how Jenkins can be tailored for MES build and testing processes, the various approaches to deploying MES components on AWS, and the best practices for deploying and monitoring these systems securely and at scale.

3.1. Setting Up Jenkins for Opcenter MES

3.1.1. Configuring Jenkins Pipelines for MES Builds

To begin with, Jenkins pipelines must be configured to support the specific build requirements of Siemens Opcenter MES. A **pipeline** in Jenkins is a scripted or declarative series of steps that define how software should be built, tested, and deployed.

For Opcenter MES, this involves setting up a Jenkins file that defines stages such as:

- **Source checkout** from a version control system like Git.
- **Environment preparation**, including installing dependencies and setting environment variables specific to MES modules.
- **Build stage**, which compiles or packages the MES components such as REST services, data models, or UI extensions.

These pipelines can be scheduled or triggered automatically based on events, such as commits to a repository. Jenkins also supports parallel execution, which is beneficial when handling multiple MES modules at once (Gupta & Sharma, 2021).

3.1.2. Automated Testing Strategies (Unit, Integration, Regression)

Automated testing ensures that MES updates do not introduce new bugs or affect existing functionalities. There are three primary types of automated tests:

- **Unit Testing**: Validates individual functions or components, such as a logic rule in an MES module. Jenkins can run unit tests using tools like JUnit or NUnit.
- **Integration Testing**: Ensures that different parts of the MES system work together as expected. For Opcenter, this might include testing interactions between the scheduling module and inventory systems.
- **Regression Testing**: Confirms that recent code changes haven't negatively impacted previously working functionality. Jenkins can re-run test suites every time a new build is generated to validate stability.

Test results are stored and visualized through Jenkins plugins, and failure notifications are sent automatically to the development team, promoting rapid debugging and quality control.

3.2. AWS Integration Approaches

3.2.1. Deploying Opcenter Components on AWS (EC2, Containers, Serverless)

There are multiple ways to deploy Opcenter MES components on AWS, depending on the specific needs of the business.

- **Amazon EC2**: Traditional Opcenter services can be hosted on EC2 instances. This method offers complete control over the environment but requires management of the underlying infrastructure.
- **Containers (ECS or EKS)**: Containerizing Opcenter modules (using Docker) allows for lightweight, consistent, and portable deployments. ECS (Elastic Container Service) or EKS (Elastic Kubernetes Service) can orchestrate these containers.
- **AWS Lambda (Serverless)**: Certain MES functionalities like notifications or simple data transformations can be offloaded to serverless functions using Lambda, improving efficiency and cost-effectiveness (AWS, 2023).

Choosing the right approach depends on performance requirements, cost considerations, and how modular the MES system is.

3.2.2. Infrastructure as Code (IaC) with AWS CloudFormation/Terraform

Infrastructure as Code (IaC) allows teams to define cloud infrastructure through code rather than manual processes. This enables version-controlled, repeatable, and auditable deployment environments.

- **AWS CloudFormation**: Allows developers to model infrastructure in YAML or JSON and deploy it across AWS regions.

- **Terraform**: An open-source IaC tool by HashiCorp that supports multi-cloud environments. It is useful if the MES system must also interface with services outside AWS.

Using IaC makes it easier to manage Opcenter dependencies, replicate environments (e.g., dev, staging, prod), and automate disaster recovery.

3.3. Secure and Scalable Deployment Models

3.3.1. Blue-Green & Canary Deployments for MES Updates

To minimize downtime and risk during updates, advanced deployment strategies can be used:

- **Blue-Green Deployment**: This method involves running two identical environments—blue (current version) and green (new version). Once the green version is tested and verified, traffic is switched from blue to green.
- **Canary Deployment**: A small percentage of users or operations are directed to the new version while the rest continue using the old one. If no issues are found, the update is rolled out to everyone.

These strategies are particularly helpful in MES systems, where downtime can severely impact manufacturing operations (Rahman et al., 2020).

3.3.2. Monitoring and Logging with AWS CloudWatch

Monitoring and logging are vital for diagnosing problems and ensuring optimal performance.

- **AWS CloudWatch**: Collects logs, metrics, and events from EC2, Lambda, ECS, and other services. For Opcenter, this helps track metrics like application errors, API response times, or memory usage.
- **CloudWatch Alarms** can automatically notify engineers or trigger auto-scaling when thresholds are exceeded. Logs can be stored for auditing and compliance purposes, which is essential for regulated industries.

4. Results and Discussion

To integrate Siemens Opcenter MES with Jenkins and deploy it on AWS securely and reliably, a structured implementation process must be followed.

4.1. Step 1: Environment Setup

4.1.1. Jenkins Server Configuration (On-Prem/Cloud)

The first step is setting up the Jenkins server. This can be done:

On-premises, for organizations that require strict data control.

In the cloud, using AWS EC2 or Jenkins-specific cloud platforms like Cloud Bees.

In both cases, Jenkins must be installed, secured with user credentials, and connected to version control systems. Plugins relevant to Docker, AWS, and testing must be configured.

4.1.2. AWS Account and IAM Roles for Secure Access

An AWS account should be set up with proper **Identity and Access Management (IAM)** roles:

Read/write access to services like EC2, ECS, and S3.

Least privilege principle to ensure users and services only access what they need.

Service accounts or access tokens must be securely stored, often in credential vaults or Jenkins secrets.

4.2. Step 2: Pipeline Automation

4.2.1. Building Opcenter Artifacts via Jenkins

Using Jenkins pipelines, the MES source code is pulled from the repository and built using automated scripts. For containerized environments, Docker files are used to package the software into containers.

Artifacts (e.g., .war or .jar files, Docker images) are then stored in a centralized repository like AWS S3 or Artifactory, ready for deployment.

4.2.2. Automated Deployment to AWS Environments

Once artifacts are ready, Jenkins can deploy them to AWS:

Using **Code Deploy** or **ECS** for container-based services.

Using **CloudFormation** or **Terraform** for environment provisioning.

Deployments can also be staged to testing environments first before moving to production.

Each step is logged and visualized for transparency.

4.3. Step 3: Validation & Rollback Mechanisms

4.3.1. Automated Testing in AWS Staging Environments

Jenkins can deploy the latest MES build to a **staging environment on AWS** for validation. Automated test suites—unit, integration, and end-to-end—are executed to confirm system integrity.

Results are reviewed before final approval for production release.

4.3.2. Rollback Strategies for Failed Deployments

If issues are detected post-deployment, **rollback strategies** include:

- **Reverting to the previous build** using Jenkins artifact history.
- **Switching environments** using Blue-Green or Canary deployment models.
- **Re-deploying infrastructure** using previous IaC templates.

These strategies ensure system stability and minimize the risk of prolonged downtime.

By implementing Jenkins pipelines and deploying Siemens Opcenter MES on AWS cloud infrastructure, organizations can achieve faster, more reliable software delivery. With careful planning around environment setup, automated testing, deployment strategies, and monitoring, businesses can reduce operational risks and increase system efficiency. Through these steps, enterprises move closer to fully integrated, agile, and cloud-native MES systems aligned with Industry 4.0.

5. Future trends and enhancements

As manufacturing systems become increasingly digitized, the integration of modern IT practices such as DevOps within Manufacturing Execution Systems (MES) continues to evolve. One of the most promising advancements lies in the use of Artificial Intelligence and Machine Learning (AI/ML) for predicting deployment failures. In traditional MES environments, deployments can be high-risk activities that, if not executed properly, can disrupt production schedules and operational continuity. With AI/ML models trained on historical deployment data, it becomes possible to anticipate potential points of failure before code even reaches production. These predictive insights can enhance deployment reliability and enable preemptive corrective actions, minimizing downtime and maintaining the integrity of manufacturing workflows. The ability to foresee and circumvent failures is not just a theoretical enhancement—it represents a practical application of intelligent systems that aligns well with the goals of continuous improvement and lean manufacturing principles (Choudhary et al., 2020).

Simultaneously, the shift towards serverless architectures is beginning to redefine how MES microservices are developed and deployed. Traditional infrastructure-heavy systems often require significant maintenance, configuration, and scaling efforts. In contrast, serverless computing abstracts away server management, allowing development teams to focus on core application logic. This model is especially attractive for MES microservices, which can benefit from the elasticity, scalability, and cost-effectiveness of serverless environments. Functions can be triggered by manufacturing events, scaling automatically based on workload, and reducing idle resource consumption. The agility offered by serverless models aligns with the DevOps ethos of rapid iteration, continuous delivery, and operational efficiency (Baldini et al., 2017).

Edge computing also emerges as a key enabler for future MES deployments, particularly in hybrid environments where responsiveness and local data processing are paramount. MES applications often operate in time-sensitive contexts where latency or connectivity issues with cloud services could impede operations. By deploying components of MES closer to the physical equipment—on the edge—systems can make real-time decisions without relying solely on centralized infrastructure. Edge computing not only reduces latency but also enhances data privacy and supports uninterrupted operation in the event of network disruptions. This distributed model, when harmonized with central cloud infrastructure through hybrid deployment strategies, supports a resilient, scalable, and future-ready MES landscape (Shi et al., 2016).

6. Conclusion

In reflecting upon the advancements within modern MES frameworks, it becomes clear that the incorporation of DevOps methodologies marks a turning point in manufacturing IT. This transformation is underpinned by key takeaways such as the acceleration of deployment cycles, the reduction in system downtime, and the enhancement of collaborative workflows between development and operations teams. The introduction of CI/CD pipelines has revolutionized the way updates and features are delivered, making MES more agile and responsive to the dynamic needs of the manufacturing floor.

The broader impact of DevOps on MES cannot be overstated. By facilitating automation, encouraging rapid feedback loops, and ensuring system stability, DevOps practices are redefining how manufacturing enterprises approach system integration and process optimization. These changes represent not merely incremental improvements but a fundamental shift in operational philosophy—one that bridges the gap between software innovation and industrial production.

Therefore, the imperative for manufacturing organizations is clear: to remain competitive in an era marked by digital transformation, they must embrace modern DevOps practices and CI/CD methodologies. By doing so, they will not only unlock greater efficiencies and resilience but also position themselves at the forefront of Industry 4.0. The integration of predictive analytics, serverless computing, and edge deployments are not distant goals but emerging realities—each representing a crucial step forward in the modernization of manufacturing execution systems.

Compliance with ethical standards

Acknowledgments

The authors would like to acknowledge the support of Thermo Fisher Scientific and Penumbra Inc. for providing the resources and environment to carry out this work. This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Disclosure of conflict of interest

Satish Kumar Nalluri declares that there is no conflict of interest related to this work. Varun Teja Bathini declares that there is no conflict of interest related to this work.

References

[1] Echezona Uzoma; Joy Onma Enyejo; Toyosi Motilola Olola. “A Comprehensive Review of Multi-Cloud Distributed Ledger Integration for Enhancing Data Integrity and Transactional Security.” Volume. 10 Issue.3, March-2025 International Journal of Innovative Science and Research Technology (IJISRT), 1953-1970, https://doi.org/10.38124/ijisrt/25mar1970Baldini, I., Castro, P., Chang, K., Cheng, P., Fink,

[2] S., Ishakian, V., ... & Suter, P. (2017). Serverless computing: Current trends and open problems. Research Advances in Cloud Computing, 1–20. https://doi.org/10.1007/978-981-10-5026-8_1

[3] Mishra, A. (2025). Ai-Powered Cybersecurity Framework for Secure Data Transmission in Iot Network. International Journal of Advances in Engineering and Management, 7(3), 05-13.

[4] Mishra, A. (2025). Ai-Powered Cyber Threat Intelligence System for Predicting and Preventing Cyber Attacks. International Journal of Advances in Engineering and Management (IJAEM), 7(2), 873-892.

[5] Choudhary, G., Kaur, G., & Saini, R. (2020). Predictive analytics using machine learning techniques for software deployment failure detection. International Journal of Advanced Computer Science and Applications, 11(3), 160–167. https://doi.org/10.14569/IJACSA.2020.0110320

[6] Shi, W., Cao, J., Zhang, Q., Li, Y., & Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5), 637–646. https://doi.org/10.1109/JIOT.2016.2579198

[7] Vel, T. (2021). AI-Driven Adaptive Authentication for Multi-Modal Biometric Systems. J. Electrical Systems, 17(1), 75-88.

[8] Madathala, H., Barmavat, B., & Thumala, S. (2023). Performance Optimization of SAP HANA using AI-based Workload Predictions. International Journal of Innovative Research in Science, Engineering and Technology, 12, 15315-15326.

[9] Amazon Web Services (AWS). (2023). AWS Documentation: EC2, Lambda, ECS, CloudWatch, and IAM. Retrieved from https://docs.aws.amazon.com/

[10] Gupta, S., & Sharma, A. (2021). Automation of CI/CD Pipelines Using Jenkins: A DevOps Approach. International Journal of Advanced Research in Computer Science, 12(4), 12–18.

[11] Rahman, M. A., Hossain, M. A., & Mamun, M. A. (2020). Deployment of Dockerized Applications in Cloud Using Jenkins. Proceedings of the International Conference on Computing Advancements (ICCA), 1–6. https://doi.org/10.1145/3377049.3377106

[12] Gupta, S., & Sharma, A. (2021). Automation of CI/CD Pipelines Using Jenkins: A DevOps Approach. International Journal of Advanced Research in Computer Science, 12(4), 12–18.

[13] Rahman, M. A., Hossain, M. A., & Mamun, M. A. (2020). Deployment of Dockerized Applications in Cloud Using Jenkins. Proceedings of the International Conference on Computing Advancements (ICCA), 1–6. https://doi.org/10.1145/3377049.3377106

[14] Amazon Web Services (AWS). (2023). AWS Documentation: EC2, ECS, Lambda, S3, and CodeDeploy. Retrieved from https://docs.aws.amazon.com/

[15] Siemens. (2023). Siemens Opcenter Manufacturing Execution System. Retrieved from https://www.plm.automation.siemens.com/global/en/products/manufacturing-operations-center/

[16] Chen, H., Zhang, Y., & Liu, H. (2021). Cybersecurity challenges and solutions in smart manufacturing: A review. Journal of Manufacturing Systems, 58, 291–304. https://doi.org/10.1016/j.jmsy.2020.07.012

[17] Kumar, R., Sharma, V., & Singh, A. (2021). Digital Transformation of Manufacturing: A Framework for MES Integration with Industry 4.0 Technologies. Procedia CIRP, 104, 1113–1118. https://doi.org/10.1016/j.procir.2021.11.187

[18] Siemens. (2023). Siemens Opcenter Execution Core MES Overview. Retrieved from https://www.plm.automation.siemens.com/global/en/products/manufacturing-operations-center/mes.html

[19] Lee, J., Kao, H. A., & Yang, S. (2020). Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment. Procedia CIRP, 16, 3–8. https://doi.org/10.1016/j.procir.2020.03.002

[20] AWS. (2024). Guidance for Deploying Siemens Opcenter Execution on AWS. Amazon Web Services. https://aws.amazon.com/solutions/guidance/deploying-siemens-opcenter-execution-foundation-on-aws/

[21] Müller, J. M., Buliga, O., & Voigt, K. I. (2022). The role of digital technologies for manufacturing execution systems in Industry 4.0. International Journal of Production Research, 60(14), 4426-4443. https://doi.org/10.1080/00207543.2021.2002965

[22] Sharma, R., Kumar, P., & Sharma, S. (2023). Implementing CI/CD for industrial systems: Challenges and solutions. Journal of Software: Evolution and Process, 35(2), e2456. https://doi.org/10.1002/smr.2456

[23] Siemens. (2024). Siemens Opcenter MES: Digital transformation for manufacturing. Siemens Digital Industries Software. https://www.plm.automation.siemens.com/global/en/products/manufacturing-execution-system-mes.html

[24] Uzoma, E., Igba, E., & Olola, T. M. (2024). Analyzing Edge AI Deployment Challenges with in Hybrid IT Systems Utilizing Containerization and Blockchain-Based Data Provenance Solutions. International Journal of Scientific Research and Modern Technology, 3(12), 125–141. https://doi.org/10.38124/ijsrmt.v3i12.408

[25] Uzoma, E., Idoko, I. P., & Enyejo, L. A. (2024). Evaluating Serverless Computing and Microservices Impact on Scalable Cloud-Native Applications and Blockchain Interoperability Frameworks. International Journal of Scientific Research and Modern Technology, 3(4), 14–17. https://doi.org/10.38124/ijsrmt.v3i4.407