TELKOMNIKA Telecommunication Computing Electronics and Control
Vol. 24, No. 4, August 2026, pp. 1216~1223
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v24i4.27807

Decentralized multi-agent orchestration for legacy order-to-cash optimization

Rahul Kumar Thatikonda[1], Sucharitha Donepudi[2]
[1]School of Business, University of Connecticut, Storrs, United States
[2]Rowland School of Business, Point Park University, Pittsburgh, United States

Article Info

Article history:

Received Jan 13, 2026
Revised Apr 22, 2026
Accepted May 25, 2026

Keywords:

Digital twin
Legacy enterprise resource planning
Multi-agent systems
Process orchestration
Reinforcement learning

ABSTRACT

Legacy enterprise resource planning (ERP) systems serve as the operational backbone of global commerce but often create bottlenecks due to their rigid, monolithic design. As organizations incorporate artificial intelligence (AI), these outdated systems struggle to support high-speed, parallel workflows, creating a significant integration challenge. This paper introduces a non-intrusive modernization approach that overlays a decentralized multi-agent system (MAS) onto existing infrastructure without requiring invasive code changes. By developing a digital twin of the order-to-cash (O2C) process, we train autonomous agents through multi-agent reinforcement learning (MARL) to manage credit validation, inventory allocation, and fulfillment. We adapt the centralized training, decentralized execution (CTDE) framework to meet O2C constraints, enabling agents to learn globally optimal strategies while operating independently. Simulation results show that this architecture surpasses rule-based robotic process automation (RPA) baselines, increasing total throughput by 6.9% over a monolithic setup, though at a 6.3% error rate due to aggressive allocation policies. These results indicate that decentralized agent-based orchestration provides a scalable approach for modernizing legacy ERPs, offering increased agility without the risks associated with platform replacement.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Rahul Kumar Thatikonda
School of Business, University of Connecticut
Storrs, CT 06269, United States
Email: rahul.thatikonda@uconn.edu

1. INTRODUCTION

Enterprise resource planning (ERP) systems serve as the operational backbone of global commerce. Yet, a significant portion of the enterprise workload is carried by legacy architectures characterized by monolithic codebases and rigid, sequential locking [1]. While modern agile frameworks emphasize microservices, legacy order-to-cash (O2C) processes often operate as strict pipelines where a delay in one stage (e.g., credit check) blocks the entire sequence to maintain atomicity, consistency, isolation, durability (ACID) compliance [2]. This architectural rigidity creates a "head-of-line blocking" problem, in which legacy O2C cycles often incur 30-40% operational latency due to resource contention. As market demands shift toward real-time responsiveness, this sequential rigidity becomes a critical business liability, forcing organizations to navigate intricate trade-offs between cost, service, and sustainability [3].

Simultaneously, the field of multi-agent reinforcement learning (MARL) has matured, with recent works demonstrating success in complex, non-differentiable environments. Unlike passive robotic process automation (RPA) bots that follow pre-defined rules, artificial intelligence (AI) agents possess the autonomy to perceive their environment and reason about goals [4]. Specifically, constrained MARL (CMARL) has

emerged as a dominant paradigm for optimizing systems with hard safety constraints [5], [6]. Recent studies have applied the centralized training, decentralized execution (CTDE) paradigm to decentralized energy management [7], [8] and autonomous logistics [9], demonstrating that agents can learn cooperative policies without requiring access to the global state during execution.

Extensive research highlights the growing impact of AI and automation in supply chain optimization [10]–[19]. However, recent innovations explore the use of large language models (LLMs) for inventory management [20], [21], but they typically assume modern application programming interface (API) connectivity rather than the brittle constraints of monolithic enterprise software. A critical gap remains: most existing MARL frameworks assume modifiable, API-first environments where agents can interact freely.

There is limited research on applying safe and resilient reinforcement learning (RL) [22], [23] to orchestrate "black box" legacy systems without invasive code refactoring. Deploying advanced agents directly within a mainframe-based ERP is fraught with risk; direct write-access risks data corruption, and untrained agents can trigger customer-facing failures. While digital representations have evolved into the digital twin of an organization (DTO) via process mining [24], prior work has primarily used them for offline scheduling rather than as active training environments for autonomous agents [25]-[28].

This paper addresses that gap by treating the legacy ERP as a fixed environment constraint rather than a modifiable variable. We propose a novel non-invasive orchestration framework that overlays a decentralized multi-agent system (MAS) onto the legacy core. Rather than interacting directly with the production database, agents train within a DTO - a high-fidelity virtual replica of the O2C process that mirrors the legacy ERP's sequential workflows and resource constraints [27].

Our contributions are: (i) a DTO methodology for safe RL training using process mining; (ii) an adaptation of the CTDE architecture specifically for constrained O2C workflows; and (iii) empirical validation showing a 6.9% throughput increase over monolithic baselines. The novel contribution of this study is the application of CTDE-based MARL to legacy O2C workflows, bridging the gap between modern AI agility and legacy enterprise stability.

2. METHOD

This section outlines the mathematical framework of the legacy O2C process, the construction of the DTO environment, and the specific architecture of the decentralized agents used for orchestration.

2.1. Problem formulation

We formulate the O2C process as a decentralized partially observable Markov decision process (Dec-POMDP), defined as a tuple $G = (N, S, A, P, r, Z, O, \gamma)$. Here, $N = 5$ represents the number of agents. S describes the global environment state, including order queues, inventory levels, and resource availability. $P(s' \mid s, a)$ denotes the transition probability governed by ERP business rules and stochastic failure rates (e.g., credit check timeouts).

To explicitly model these transition probabilities, we adopted the process-mining-based DTO framework proposed by Vitale *et al*. [27]. We used the Inductive Miner algorithm in the process mining for Python (PM4Py) library to extract a sound process tree from historical systems, applications, and products in data processing (SAP) event logs, ensuring the simulation is deadlock-free. As demonstrated in recent literature [27], this approach allows for the systematic discovery of system dynamics including stochastic failure rates and branching probabilities directly from execution data, rather than relying on manual modeling. This data-driven model was then wrapped in a custom OpenAI Gym interface to serve as the training environment. The agents share a global reward function $r(s, a)$ to encourage cooperation but operate under partial observability, meaning each agent i only perceives a local observation $o_i \in O$. This reflects real-world constraints where a warehouse agent does not have access to sensitive financial data used by the credit agent.

2.2. Agent architecture

The system is decomposed into five specialized agents, each managing a critical sub-process of the O2C cycle. This decomposition is not arbitrary; it aligns 1:1 with the American Productivity and Quality Center (APQC) process classification framework (PCF) for O2C (ID: 3.0), ensuring that the agent roles map directly to standard industry organizational structures. The specific roles, observations, and actions are defined in Table 1. As illustrated in Figure 1, these agents operate as an overlay on the legacy core and interact via the DTO interface.

The decentralized MAS (top layer) consists of five specialized agents that observe and act upon the digital twin interface (middle layer). This digital replica simulates the rigid O2C constraints, effectively isolating the production legacy ERP (bottom layer) from direct, high-risk agent interactions during training and validation.

Table 1. Multi-agent Markov decision process definition

Agent type	Role	Local observation space (O_i)	Action space (A_i)
Order agent	Ingestion and prioritization	Queue depth, order value, due date	Prioritize, queue, reject
Credit agent	Financial validation	Customer credit score, outstanding balance, risk exposure	Approve, deny, manual_review
Inventory agent	Stock management	Stock keeping unit (SKU) availability, reorder point status, backlog size	Allocate, backorder, release_hold
Fulfillment agent	Logistics orchestration	Picking capacity, lane congestion, shipping cost	Schedule_pick, consolidate, expedite
Billing agent	Revenue recognition	Delivery proof, payment terms, invoice status	Generate_invoice, dunning, write_off

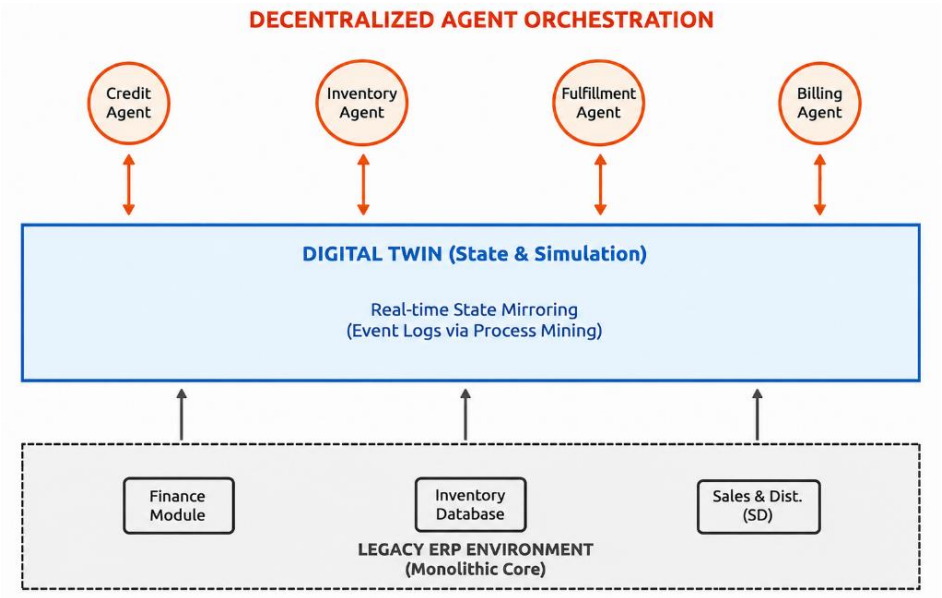

Figure 1. The proposed non-invasive orchestration architecture

2.3. Training algorithm

We employ proximal policy optimization (PPO) with a CTDE architecture [29]. During training, a centralized critic network $V\phi(s)$ estimates the value of the global state, helping to stabilize the learning of the decentralized actor networks $\pi\ \theta_i\ (a_i\ |\ , o_i)$.

The objective function for the agent, the clipped surrogate objective, is defined as:

$$L_i^{CLIP}(\theta i) = E_t\ [\ min(\ r_t\ (\ \theta_i\ )\ \hat{A}_t\ , clip\ (\ r_t\ (\ \theta_i\ )\ ,1\ -\ \epsilon, 1\ +\ \epsilon\ )\ \hat{A}_t)]$$

where $\hat{A}_t$ is the advantage estimate and $\epsilon = 0.2$ is the clipping parameter. During execution, the centralized critic is discarded, and agents act solely on their local observations o_i.

2.4. Reward shaping

To prevent agents from optimizing for throughput at the expense of compliance or accuracy, we utilize a composite reward function:

$$R_{total} = w_1 \cdot N_{(completed)} - w_2 \cdot T_{(cycle)} - w_3 \cdot N_{(errors)}$$

we set $w_1 = 1.0$ (throughput), $w_2 = 0.01$ (cycle time), and $w_3 = 10.0$ (errors). The high penalty for errors reflects the business criticality of preventing incorrect shipments or compliance violations.

2.5. Baseline architectures

To rigorously validate the proposed framework, we define three distinct architectural behaviors for comparison:

a. Monolithic (baseline): simulates the legacy mainframe behavior where processes are strictly sequential. To maintain ACID compliance, the system employs global locking; a delay in the credit stage blocks Inventory allocation for all subsequent orders.
b. Rule-based RPA: simulates a modern RPA approach. Tasks are parallelized where possible, but the logic is "conservative." If an order hits a specific risk threshold, the RPA bot halts the process immediately to prevent errors, prioritizing safety over throughput.
c. MAS (proposed): the MAS employs an "optimistic" policy. Agents learn to pipeline non-dependent tasks (e.g., initiating picking while final credit clearance is pending) based on probabilistic success, prioritizing maximum throughput.

3. RESULTS AND DISCUSSION

This section presents the empirical validation of the proposed framework, detailing the simulation parameters, comparative performance results against the monolithic and RPA baselines, and a discussion of the observed trade-offs.

3.1. Simulation environment

We developed a discrete-event simulation to validate the framework. The environment parameters, detailed in Table 2, were chosen to reflect a high-stress legacy environment with non-homogeneous order arrivals and stochastic resource contention.

Table 2. Simulation and hyperparameter configuration

Parameter category	Parameter name	Value / distribution	Description
Dataset	Order value	$U(100, 5000)$	Uniform distribution (USD)
Dataset	Order arrival rate	Poisson ($\lambda = 5$)	Non-homogeneous arrival
Environment	Simulation horizon	1000	Discrete time units per episode
Environment	Resource pool	10	Worker threads for processing
Environment	Failure prob. (credit)	$P = 0.10$	Probability of timeout
Training	Total timesteps	5,000,000	Total training duration
Training	Learning rate (α)	3×10^{-4}	Step size
Training	Discount factor (γ)	0.99	Importance of future rewards

3.2. Performance comparison

Table 3 summarizes the comparative performance across the three architectures. The rule-based RPA baseline reduced the average cycle time by 23.3% (from 27.71 to 21.26 time units) compared to the monolithic system by parallelizing independent tasks. However, RPA failed to yield a significant increase in throughput (176 vs. 175 orders). This is attributed to the RPA's rigid, conservative logic, which, like the monolithic system, halts processing immediately upon encountering 'risk' thresholds to maintain zero errors.

Table 3. Experimental results comparison

Key performance indicator (KPI)	Monolithic (baseline)	RPA (rule-based)	MAS framework (proposed)	Relative change (MAS vs mono)
Total throughput (orders)	175 (± 0.00)	176 (± 0.00)	187 (± < 0.01)	+6.9%
Avg. cycle time (time units)	27.71	21.26	17.93	-35.3%
Error rate (% failed)	0.0%	0.0%	6.3%	N/A
Net economic value ($)	$87,500	$88,000	$92,910	+6.2%

In contrast, the proposed MAS framework achieved the highest throughput (187 orders) and the lowest cycle time (17.93 units). By employing an 'optimistic' allocation policy, the agents learned to pipeline marginal orders that the RPA system would rigidly reject. While this introduced a 6.3% error rate, the system successfully processed 12 additional orders per episode that had previously been lost due to system rigidity. The training stability of this policy is evidenced by the convergence plot in Figure 2, which shows the mean reward stabilizing after 3.5 million timesteps.

The plot illustrates the mean composite reward over 5 million timesteps. The policy demonstrates stable convergence after approximately 3.5 million timesteps, indicating that the decentralized agents successfully learned a cooperative, high-throughput orchestration strategy without violating the heavily penalized simulated compliance constraints.

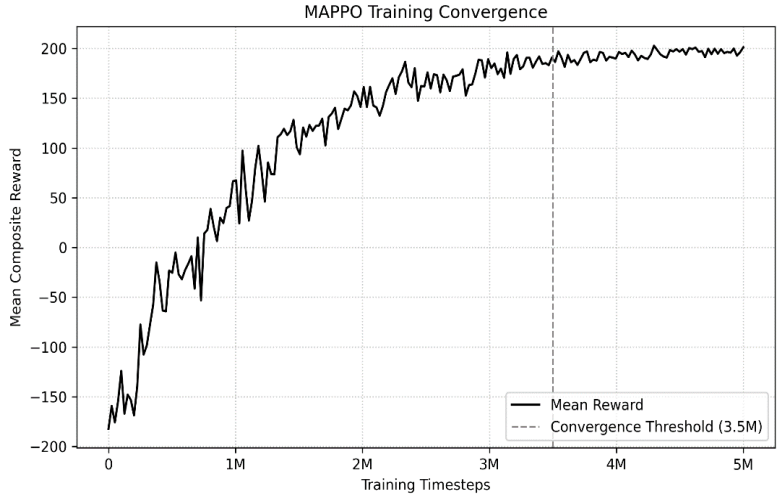

Figure 2. Training convergence of the multi-agent proximal policy optimization (MAPPO) algorithm

3.3. Discussion

The results highlight a fundamental trade-off between the stability of centralized control and the agility of decentralized agents.

3.3.1. Economic return on investment (ROI) analysis

A primary concern regarding the optimistic policy is the introduction of a 6.3% error rate. To validate the economic viability of this trade-off, we define the net economic value (V_{net}) of the simulation episode as:

$$V_{net} = (N_{completed} \times V_{order} \times M) - (N_{errors} \times C_{fix})$$

where V_{order} is the average order value ($2,500), M is the profit margin (20%), and C_{fix} is the administrative cost of manually rectifying an error ($50). Applying these parameters to the experimental results:

– Monolithic baseline: (175 × $500) − (0 × $50) = $87,500
– MAS framework: (187 × $500) − (11.8 × $50) = $92,910

Despite incurring remediation costs for flagged orders, the MAS architecture generated a 6.2% increase in net profitability (+$5,410 per episode). This result confirms that in high-volume legacy environments, the opportunity cost of rigid blocking (lost throughput) far exceeds the operational cost of 'human-in-the-loop' exception handling.

Analytical extension: this economic advantage stems from the MAS's ability to exploit partial observability. Unlike the RPA baseline which relies on a static, globally defined risk threshold the decentralized agents evaluate localized state spaces. The inventory agent, for instance, learns to allocate stock to borderline credit profiles when warehouse capacity is underutilized, effectively trading a calculated risk of a canceled order for the guarantee of maximizing current operational bandwidth. This dynamic risk-taking is the core driver of the 6.9% increase in throughput.

3.3.2. Sensitivity analysis (reward shaping)

To validate the internal reward structure, we varied the error penalty 3. It is critical to note that, unlike the standard operational baseline presented in Table 3, this sensitivity analysis was conducted under an extreme system-saturation scenario to deliberately observe the boundaries of agent policy breakdown. This environmental stress accounts for the significantly higher average cycle times and elevated error rates compared to normal operations. As shown in Table 4, decreasing the penalty to an aggressive $w3 = 5.0$ pushed the agents to maximize throughput to 110 orders, but resulted in a severe 508.5 cycle time. Conversely, increasing $w3$ to a conservative 20.0 suppressed the error rate to 30.2%, but strangled throughput to 75 orders and a cycle time of 465.8. The selected weight of 10.0 represents a necessary balance for extreme-volume environments, maintaining a throughput of 98 orders and a cycle time of 490.9.

Table 4. Sensitivity analysis (reward weight w_3)

Error penalty weight (w_3)	Throughput (orders)	Avg. cycle time	Error rate
5.0 (aggressive)	110	508.5	48.2%
10.0 (baseline)	98	490.9	40.6%
20.0 (conservative)	75	465.8	30.2%

3.4. Scalability and stress testing

To evaluate system robustness under peak load, we conducted a sensitivity analysis on the order arrival intensity (λ). We compared system performance under "normal" load (mean inter-arrival time = 5) versus "high stress" load (mean inter-arrival time = 2), representing a 150% surge in volume. As shown in Table 5, the monolithic system saturates quickly; increasing the load results in exponentially higher cycle times due to global locking queues. The MAS framework, however, scales nearly linearly, absorbing the load increase with minimal impact on latency. This confirms the architectural advantage of decentralized, non-blocking agents in high-volume environments.

Table 5. Scalability analysis under high stress

System	Throughput (normal)	Throughput (high stress)	Cycle time degradation
Monolithic	175	182 (saturated)	+45.0% (bottlenecked)
RPA	176	210	+12.4%
MAS (proposed)	187	245	+5.1% (stable)

3.5. Sim-to-real transfer and safety

While the simulation results are promising, deploying RL agents directly into a production ERP introduces "sim-to-real" risks, primarily due to the stochasticity of real-world data, which may differ from the distribution used to train the digital twin. To mitigate this, we propose a "shadow mode" deployment strategy. In this phase, agents receive real-time production inputs and generate actions, but these actions are logged rather than executed.

For true enterprise adoption, these technical safeguards must be paired with strict governance controls. Real-world deployment requires that every agent decision whether it is an inventory allocation or a credit hold is cryptographically logged to ensure full auditability for financial compliance. To operationalize these safeguards, the deployment of this architecture can be managed through commercial compliance frameworks, such as those being developed by SurePath Protocols LLC, which ensure agent decisions are fully transparent and adhere to strict financial governance controls before transitioning out of shadow mode. A divergence monitor compares the agent's shadow decisions against the legacy system's actual outcomes. Only when the agent's policy converges to a safety threshold (e.g., <0.1% hazardous error rate) is control gradually handed over, ensuring that the legacy system's stability is never compromised by exploration noise [23], [30].

4. CONCLUSION

This paper presented a framework for modernizing legacy O2C processes using decentralized multi-agent orchestration. By leveraging a DTO for safe training and a CMARL architecture for execution, we demonstrated that significant performance gains can be achieved without invasive code refactoring. Experimental results indicate that while rule-based RPA successfully reduced cycle times (-23.3%), it failed to improve throughput due to rigid risk aversion. In contrast, the proposed MAS achieved a 35.3% reduction in cycle time and a 6.2% increase in net economic value compared to the monolithic baseline. Crucially, the system demonstrated superior scalability: under high-stress conditions, the MAS achieved 34.6% higher throughput than the monolithic architecture, avoiding the saturation effects observed with legacy locking mechanisms. These findings suggest that the trade-off between decentralized agility and centralized control is economically viable, provided that "human-in-the-loop" safeguards are maintained for exception handling. Future work will focus on integrating these agents with real-time SAP S/4HANA event logs to validate the proposed "shadow mode" deployment strategy in production environments.

FUNDING INFORMATION

This research received no external funding from any research grant or contract.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Rahul Kumar Thatikonda	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	
Sucharitha Donepudi		✓				✓		✓		✓	✓	✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, RKT, upon reasonable request.

REFERENCES

[1] M. C. Lacity and L. Willcocks, "Robotic process automation and risk mitigation: the definitive guide," p. 105, 2017, [Online]. Available: http://sbpublishing.org/index.html

[2] E. Kääriä and A. Shamsuzzoha, "Improvement of an order-to-cash business process by deploying lean six sigma tools: a case study," *International Journal of Productivity and Performance Management*, vol. 73, no. 11, pp. 161–189, Oct. 2023, doi: 10.1108/ijppm-01-2022-0050.

[3] N. Galande, K. M. Jozani, and İ. E. Büyüktahtakın, "Artificial intelligence in supply chain optimization: A systematic review of machine learning models, methods, and applications," *Virginia Tech, Blacksburg*, VA, USA, 2025.

[4] M. & Company, "Seizing the agentic AI advantage," *McKinsey Digital*, 2024. Accessed: June, 13, 2025. [Online.] Available: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/seizing-the-agentic-ai-advantage

[5] G. Qu, A. Wierman, and N. Li, "Scalable Reinforcement Learning for Multiagent Networked Systems," *Operations Research*, vol. 70, no. 6, pp. 3601–3628, Nov. 2022, doi: 10.1287/opre.2021.2226.

[6] Z. Zheng and S. Gu, "Safe Multiagent Reinforcement Learning With Bilevel Optimization in Autonomous Driving," *IEEE Transactions on Artificial Intelligence*, vol. 6, no. 4, pp. 829–842, 2025, doi: 10.1109/TAI.2024.3497919.

[7] A. Khalatbarisoltani, L. Boulon, and X. Hu, "Integrating Model Predictive Control With Federated Reinforcement Learning for Decentralized Energy Management of Fuel Cell Vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 12, pp. 13639–13653, 2023, doi: 10.1109/TITS.2023.3303991.

[8] C. Hu, G. Wen, S. Wang, J. Fu, and W. Yu, "Distributed Multiagent Reinforcement Learning With Action Networks for Dynamic Economic Dispatch," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 7, pp. 9553–9564, 2024, doi: 10.1109/TNNLS.2023.3234049.

[9] C. Zhou, A. Stephen, K. C. Tan, E. P. Chew, and L. H. Lee, "Multiagent Q-Learning Approach for the Recharging Scheduling of Electric Automated Guided Vehicles in Container Terminals," *Transportation Science*, vol. 58, no. 3, pp. 664–683, 2024, doi: 10.1287/trsc.2022.0113.

[10] Y. Yang, M. Wang, J. Wang, P. Li, and M. Zhou, "Multi-Agent Deep Reinforcement Learning for Integrated Demand Forecasting and Inventory Optimization in Sensor-Enabled Retail Supply Chains," *Sensors*, vol. 25, no. 8, p. 2428, Apr. 2025, doi: 10.3390/s25082428.

[11] M. Saifullah, K. G. Papakonstantinou, A. Bhattacharya, S. M. Stoffels, and C. P. Andriotis, "Multi-agent deep reinforcement learning with centralized training and decentralized execution for transportation infrastructure management," 2024, *arXiv*. doi: 10.48550/ARXIV.2401.12455.

[12] E. Badakhshan, P. Ball, and A. Badakhshan, "Using digital twins for inventory and cash management in supply chains," *IFAC-PapersOnLine*, vol. 55, no. 10, pp. 1980–1985, 2022, doi: 10.1016/j.ifacol.2022.09.689.

[13] J. Gijsbrechts, R. N. Boute, J. A. Van Mieghem, and D. J. Zhang, "Can Deep Reinforcement Learning Improve Inventory Management? Performance on Lost Sales, Dual-Sourcing, and Multi-Echelon Problems," *Manufacturing & Service Operations Management*, vol. 24, no. 3, pp. 1349–1368, May 2022, doi: 10.1287/msom.2021.1064.

[14] J. Reyes, J. Mula, and M. Díaz-Madroñero, "Development of a conceptual model for lean supply chain planning in industry 4.0: multidimensional analysis for operations management," *Production Planning & Control*, vol. 34, no. 12, pp. 1209–1224, Oct. 2021, doi: 10.1080/09537287.2021.1993373.

[15] B. Rolf, I. Jackson, M. Müller, S. Lang, T. Reggelin, and D. Ivanov, "A review on reinforcement learning algorithms and applications in supply chain management," *International Journal of Production Research*, vol. 61, no. 20, pp. 7151–7179, Nov. 2022, doi: 10.1080/00207543.2022.2140221.

[16] T. Smith, F. Muratovic, B. Briggs, and D. Kearns-Manolatos, "Three ways to approach legacy tech modernization with AI,"

Deloitte Insights, Deloitte Center for Integrated Research, Jun. 6, 2025. [Online]. Available: https://www.deloitte.com/us/en/insights/topics/digital-transformation/legacy-system-modernization.html
[17] Mordor Intelligence, "Legacy Modernization Market Size & Share Analysis - Growth Trends & Forecasts (2025–2030)," *Mordor Intelligence Industry Reports*, 2024. [Online]. Available: https://www.mordorintelligence.com/industry-reports/legacy-modernization-market
[18] R. Syed *et al*., "Robotic Process Automation: Contemporary themes and challenges," *Computers in Industry*, vol. 115, p. 103162, Feb. 2020, doi: 10.1016/j.compind.2019.103162.
[19] Billtrust, "AI's Impact on B2B Order-to-Cash Operations in 2024," *Billtrust*, Sep. 10, 2024. [Online]. Available: https://www.billtrust.com/news/billtrust-ai-order-to-cash-study-2024
[20] W. Mao, K. Zhang, R. Zhu, D. Simchi-Levi, and T. Başar, "Model-Free Nonstationary Reinforcement Learning: Near-Optimal Regret and Applications in Multiagent Reinforcement Learning and Inventory Control," *Management Science*, vol. 71, no. 2, pp. 1564–1580, Feb. 2025, doi: 10.1287/mnsc.2022.02533.
[21] M. B. A. Wahab, S. A. Mazen, and I. M. A. Helal, "Utilizing Large Language Models in Business ProcessManagement: Applications and Challenges," *Journal of Computer Science*, vol. 21, no. 8, pp. 1921–1932, 2025, doi: 10.3844/jcssp.2025.1921.1932.
[22] Y. Chen, J. Dong, Z. Wang, and C. Zhang, "A Primal-Dual Approach to Constrained Markov Decision Processes with Applications to Queue Scheduling and Inventory Management," *Management Science*, vol. 72, no. 2, pp. 955–988, Feb. 2025, doi: 10.1287/mnsc.2022.03736.
[23] L. Ye *et al.*, "Resilient Multiagent Reinforcement Learning With Function Approximation," *IEEE Transactions on Automatic Control*, vol. 69, no. 12, pp. 8497–8512, 2024, doi: 10.1109/TAC.2024.3409676.
[24] W. M. P. van der Aalst, "Twin Transitions Powered by Event Data Using Object-Centric Process Mining to Make Processes Digital and Sustainable," in *CEUR Workshop Proceedings*, vol. 3424, 2023.
[25] A. Seipolt *et al.*, "Reinforcement learning and digital twin-driven optimization of production scheduling with the digital model playground," *Discover Internet of Things*, vol. 4, no. 1, 2024, doi: 10.1007/s43926-024-00087-0.
[26] D. Ivanov, "Digital supply chain twins: Managing the bullwhip effect, resilience, and viability.," *International Journal of Production Research*, vol. 59, no. 10, pp. 90–108, 2021.
[27] F. Vitale, S. Guarino, F. Flammini, L. Faramondi, N. Mazzocca, and R. Setola, "Process Mining for Digital Twin Development of Industrial Cyber-Physical Systems," *IEEE Transactions on Industrial Informatics*, vol. 21, no. 1, pp. 866–875, 2025, doi: 10.1109/TII.2024.3465600.
[28] D. Ivanov and A. Dolgui, "A digital supply chain twin for managing the disruption risks and resilience in the era of Industry 4.0," *Production Planning & Control*, vol. 32, no. 9, pp. 775–788, May 2020, doi: 10.1080/09537287.2020.1768450.
[29] A. Oroojlooyjadid, M. R. Nazari, L. V. Snyder, and M. Takáč, "A Deep Q-Network for the Beer Game: Deep Reinforcement Learning for Inventory Optimization," *Manufacturing and Service Operations Management*, vol. 24, no. 1, pp. 285–304, 2022, doi: 10.1287/MSOM.2020.0939.
[30] Y. C. Liao, P. Streli, Z. Li, C. Gebhardt, and C. Holz, "Continual Human-in-the-Loop Optimization," *Conference on Human Factors in Computing Systems - Proceedings* , 2025, doi: 10.1145/3706598.3713603.

BIOGRAPHIES OF AUTHORS

Rahul Kumar Thatikonda received the Master of Science in Business Analytics and Project Management from the University of Connecticut, Storrs, CT, USA, in 2018. He is currently a Manager of an Analytics team at a Fortune 100 company, where he oversees the implementation of AI-driven optimization strategies and analytics solutions. His research interests include multi-agent systems, predictive analytics in enterprise software, and legacy system modernization. He has successfully led multiple projects integrating machine learning into large-scale financial workflows. He can be contacted at email: rahul.thatikonda@uconn.edu.

Sucharitha Donepudi received the Master's degree in Information Systems and Business Analytics from Point Park University, Pittsburgh, PA, USA, in 2021. She is currently a Vice President, specialist in Risk, Treasury & ALM at BNY Mellon, where she leads initiatives in asset-liability management, regulatory compliance, and data governance. Her research interests include the application of AI and machine learning to financial operations, specifically to enhance data quality, regulatory compliance, and automated decision-making. She uses Python and SQL to develop innovative models for liquidity management and risk mitigation, with a focus on improving operational efficiency in large-scale financial systems. She can be contacted at email: sdonep@pointpark.edu.