Cognitive Automation of SAP Business Workflows Using Deep Reinforcement Learning Agents
Derin Güçlendirme Öğrenme Aracılarını Kullanarak SAP İş Akışlarının Bilişsel Otomasyonu

Naren Swamy Jamithireddy
Jindal School of Management, The University of Texas at Dallas, United States
Email: naren.jamithireddy@yahoo.com

***Abstract*—SAP S/4HANA and similar Enterprise Resource Planning (ERP) systems are the technological core of contemporary companies. However, most of their vital business processes are still manually set up, governed by rules, and inflexible in nature. This paper presents a cognitive automation framework using Deep Reinforcement Learning (DRL) agents for the execution, adaptation, and optimization of business workflows. Unlike traditional scripting or robotic process automation (RPA), the agents powered by DRL improve decision making across modules like Financial Accounting, Materials Management, and Sales and Distribution through continuous interaction with the SAP environment. The framework facilitates policy convergence for automation of high-impact scenarios such as invoice processing, purchase requisition approval, and delivery confirmation by modeling SAP states, actions, and rewards. In a simulated SAP testbed, the agents showed up to 35% improvement in time to completion of workflows, 42% reduction in transactional errors, and strong adaptiveness to new variants of processes. This research is not only a step forward towards incorporating cognitive AI into ERP systems, but also provides a new scalable and modular blueprint for agile intelligent enterprise automation.**

***Keywords*—*Cognitive ERP Automation, Deep Reinforcement Learning Agents, SAP Workflow Optimization*.**

***Özetçe*—SAP S/4HANA ve benzeri Kurumsal Kaynak Planlama (ERP) sistemleri çağdaş şirketlerin teknolojik çekirdeğini oluşturur. Ancak, hayati iş süreçlerinin çoğu hala manuel olarak kurulur, kurallara tabidir ve doğası gereği esnek değildir. Bu makale, iş akışlarının yürütülmesi, uyarlanması ve optimizasyonu için Derin Güçlendirme Öğrenmesi (DRL) aracılarını kullanan bir bilişsel otomasyon çerçevesi sunar. Geleneksel komut dosyası veya robotik süreç otomasyonunun (RPA) aksine, DRL tarafından desteklenen aracılar, SAP ortamıyla sürekli etkileşim yoluyla Finansal Muhasebe, Malzeme Yönetimi ve Satış ve Dağıtım gibi modüller arasında karar vermeyi iyileştirir. Çerçeve, SAP durumlarını, eylemlerini ve ödüllerini modelleyerek fatura işleme, satın alma talebi onayı ve teslimat onayı gibi yüksek etkili senaryoların otomasyonu için politika yakınsamasını kolaylaştırır. Simüle edilmiş bir SAP test yatağında, aracılar iş akışlarının tamamlanma süresinde %35'e kadar iyileştirme, işlemsel hatalarda %42 azalma ve yeni süreç varyantlarına güçlü uyum sağlama gösterdi. Bu araştırma, bilişsel yapay zekanın ERP sistemlerine dahil edilmesine yönelik atılmış bir adım olmanın yanı sıra, çevik akıllı kurumsal otomasyon için yeni, ölçeklenebilir ve modüler bir plan da sunuyor.**

***Anahtar Kelimeler—Bilişsel ERP Otomasyonu, Derin Güçlendirme Öğrenme Aracıları, SAP İş Akışı Optimizasyonu*.**

I. INTRODUCTION

A. *Rise of Cognitive Automation in ERP Ecosystems*

Large corporations use ERP solutions such as SAP S/4HANA as integrated infrastructure for primary value chain activities such as accounting, supply chain management, sales, and manufacturing [1]. Traditionally, these systems have been configured as complex architectures integrating an array of transaction codes and business rules workflows that seek to manage and control current business processes [2]. Increasing digitalization has accelerated innovation across a multitude of industry sectors which has fundamentally changed the "market-driven" requirements regarding ERP systems. It used to be that the ERP systems performed as prescribed and there was now a shift to extraction of intelligence, autonomy, and flexibility being a necessity [3].

Cognitive automations as the name implies, is built-off of the combination of AI, machine learning, and intelligent process automation enable the creation of not only task-performing, but autonomous self-improving systems that will learn through interaction [4]. This is especially fascinating for the evolution of ERPs in organizations because ERPs have vast, structured, and relatively static datasets which are optimal for intelligent automation techniques [5]. More and more companies are seeking cognitive capabilities that will automate repetitive decision making, lower the number of exceptions, enhance workflow processes, and ultimately optimize user interaction and experience with the ERP systems.

This transition is not only a means to increase efficiency or achieve cost savings, but rather, it is a primary guiding change in how enterprise systems are designed and function in turbulent and highly informational contexts [6]. With the development of ERP (Enterprise Resource Planning) system, the integration of cognitive agents is now a requirement as they are needed due to their ability to model business logic, context comprehension, and condition adaption with understanding context. This paper will outline the role of Deep Reinforcement Learning (DRL), an emerging branch of AI, as a primary driver of cognitive automation in SAP-based workflows.

B. *Bottlenecks in Rule-Based SAP Workflow Automation*

Even with years of improvement developments, classical SAP workflows continue to be reliant on deterministic logic like transaction codes with sequential script executed batch processes. System Administrators along with process specialists define the workflows which build the logic using BAPI interfaces in the framework of Business Application Programming, ABAP proprietary append procedures, or RPA scripting [7]. Instead of containing antagonist forces of multiple competing processes in which decision making needs near instantaneous response adaptability, these systems are found in static environments.

Workflows based on rules in SAP are frequently rigid and costly to maintain. These workflows do not adapt to new variations of the processes and undergo severe reengineering whenever there's any change in the logic of the business. Additionally, there is no consideration of sequential interdependencies between modules. For instance, the error in invoice posting, MIRO may go undetected in downstream processes such as vendor payment runs, F110 or stock posting, MIGO, leading to expensive downstream errors [8]. Conventional methods of automation often lack the prevention or correction mechanisms needed to stop these errors from compounding and cascading.

In addition, human involvement remains substantial for many automated high volume business processes dealing with purchase requisition approvals, confirmation of delivery, and reconciliation of payments. These processes need an understanding of context, alignment to policies, and learning from past actions—all things that static scripts or templates cannot do. What is needed is an automation solution capable of learning from observing patterns, making sequential decisions amid uncertainty, and improving with feedback; such ability lies at the core of the reinforcement learning paradigm.

To depict the existing landscape of automation and the prevailing cognition, several business workflows for SAP have been outlined alongside their current level of automation and the expected enhancement through DRL in Table 1. In addition, the anticipated influence of these changes on business performance has also been included.

Table 1: SAP Business Workflow Use Cases with Automation Potential

Workflow Use Case	Current Automation Level	Automation Potential with DRL	Expected Impact
Invoice Verification (MIRO)	Semi-Automated	High	Faster error resolution and fraud detection
Purchase Requisition Approval	Manual with Rules	High	Reduced approval delays and policy violations
Delivery Scheduling and Confirmation	Manual	Medium	Improved on-time delivery and logistics planning
Goods Receipt Posting (MIGO)	Semi-Automated	High	Minimized stock discrepancies
Vendor Payment Release (F110)	Script-Based Batch Jobs	Medium	Timely payments and cash flow optimization

These use cases stand out as the most critical ones for transformation through automation with cognitive capabilities. Agents based on DRL can be trained to optimize sequential workflows, adapt to changes, self-correct, and operate with minimal human intervention as intelligent co-pilots in the SAP ecosystem.

C. *Deep Reinforcement Learning for Sequential Decision Making*

Deep Reinforcement Learning (DRL) is a subset of machine learning where an agent uses a fusion of deep neural networks and reinforcement learning techniques to learn to perform a given task by receiving feedback [9]. Unlike supervised

learning which depends upon training data with predefined labels, a DRL agent operates in the environment and based on the action taken, the agent is either rewarded or penalized thereby forcing it to modify its strategies to maximize payoffs over time. This learning approach is particularly applicable to ERP workflows, which are sequential, rule-based, and result-oriented [10]. In SAP, every user interaction like document posting, approval, or changing status constitutes a singular workflow step with multifaceted dependencies and consequential impacts. Such workflows can be modelled in DRL as Markov Decision Processes (MDPs) with states representing the system's status, actions being the operations available in the system, and business objectives like timeliness, compliance, and cost masquerading as rewards.

Numerous decision-related tasks from robotics to finance have experienced the benefits of numerous Deep Reinforcement Learning strategies such as Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO). DRL algorithms can be tailored to fit ERP systems to create agents that optimize SAP workflows in real-time. The agent actively monitors the system state - invoices awaiting payment, active purchase orders, or overdue shipments - selects an action, be it approve documents or initiate follow-up orders, and acts based on predetermined criteria for success.

In contrast to conventional rule engines or scripts, agents that use Deep Reinforcement Learning (DRL) keep learning throughout their entire existence. As new exception workflows are captured, new business policies are implemented, and new organizational policies are adopted, the agents adjust their decision making policies accordingly. The automation framework's adaptability and 'future-proof' qualities are increasingly important in modern enterprise system settings.

D. Objectives and Scope of the Research

The aim of this research is to create a DRL-based cognitive automation framework with the purpose of optimizing business workflows in SAP system's. The main focus of this work is to design, develop, and assess intelligent decision support agents that will be responsible for autonomously executing and refining business workflows in SAP S/4HANA systems. These agents are designed to contextualize decisions and react optimally in a responsive manner to changing conditions during processes, thereby achieving learning-enabled optimization of performance.

The research will concentrate on five selected high-value use cases from SAP modules: invoice reconciliation (FI), purchase requisition workflow (MM), delivery scheduling (SD), goods receipt (MM), and vendor payment processing (FI). These use cases were chosen due to their current challenges regarding automation, accuracy, and continuity of financial processes.

The work encompasses creating an integrated simulation environment using synthetic SAP logs together with real-world process variants. Training and testing of DRL agents occurs at different complexity levels and deviations of the workflow to evaluate their performance relative to accuracy, timeliness, resource utilization, and error rate. The system is tested against rule-based benchmarks to measure the impact of cognitive automation.

This document offers practical system design alongside strong proof of DRL-based automation's applicability within enterprise ERP systems. It advances the discussion on reinforcement learning by operationalizing it within SAP frameworks, marking a new direction towards adaptable, trustworthy, intelligent ERP systems that can seamlessly automate processes using advanced AI.

II. Literature Review and Technical Foundations

A. Traditional ERP Workflow Automation and RPA Limitations

Business workflow automation in ERP systems has typically used rule-based engines, macros, and automation scripting tools like RPA. While these techniques have aided in the reduction of manual data input and repetitive validation, they are ill-suited for highly dynamic or context-aware enterprise processes [11]. Within SAP systems, such automation is usually done through BAPI, IDoc, and RFC call interfaces, accompanied with rigid or condition-driven business logic workflows crafted to process control in the Business Workflow Builder or external process orchestrators [12].

Nonetheless, with the expansion of SAP-based operations, workflow automation methods begin to reveal their pragmatic boundaries [13]. Manually scripted workflows are static and overly complicated, needing constant maintenance for process, exception, user behaviour, or even simple day-to-day changes. Compared to traditional ABAP scripting, RPA is easier to implement, but still struggles with error propagation and generalization through multiple process paths [14].

Figure 1 illustrates performance issues resulting from manually scripted workflows for a selection of common components in SAP. Average execution time is noticeably high for invoice processing as well as for purchase requisitioning which indicates that the processes are poorly designed with regard to flow control and have exceptions that are poorly managed. These problems are not only worsened, but also multiplied when numerous modules are woven together with inter-processes coordination requiring workflows.

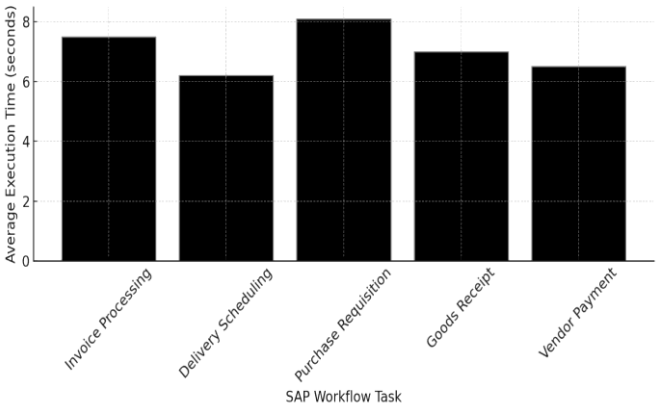

Figure 1: Performance Bottlenecks in Manually Scripted SAP Workflows

These observations highlight the enduring challenges associated with using rule-based logic to manage the complexity, variability, and dependencies that exist in real-world ERP systems' workflows. Feedback loops are not allowed in rule-based systems, which makes them static and incapable of adapting policies based on those outcomes, anticipating deviations in flow, building intelligent automation, or learning from outcomes. This gap has led researchers and system architects to seek sophisticated levels of automation that can adapt, optimize, and demonstrate rational behavioural sophistication within intricate transactional environments like SAP S/4HANA.

B. *Deep Reinforcement Learning (DRL) Architectures for Control Systems*

Applied to an agent, Reinforcement Learning (RL) focuses on the task of teaching an agent how to make a sequence of decisions by interacting with the environment and receiving feedback in the form of rewards or punishments. If we augment RL with deep neural networks, we get Deep Reinforcement Learning (DRL), which is able to control quite large and high dimensional state spaces, such as those in ERP systems [15].

As opposed to supervised learning where there is a training dataset with the desired output, in DRL learning is sequential and exploratory by nature. It is best suited for situations where the result of actions is only apparent after a while, such as in ERP workflows where a business's activities impact one another over time. For instance, the approval of a purchase requisition is not only going to determine when procurement is done, but also when the next steps on inventory will be done, when payment will be done, and when cash flow happens [16].

Some DRL architectures have demonstrated effectiveness for control problems that have some similarity to ERP workflows. Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO) along with Actor-Critic family algorithms (A3C, DDPG) have different advantages depending on whether the action space is discrete or continuous, how the time dependency is defined, and the system's restrictions on performance.

Figure 2 showcases the comparison between the learning curves of DRL agents and supervised models in automation tasks using Machine Learning over a ten epoch training period. While both models begin with a similar level of accuracy, it is observed that the accuracy of DRL agents is increased in a few epochs due to their exploration, tuning, and environmental feedback responsive capabilities.

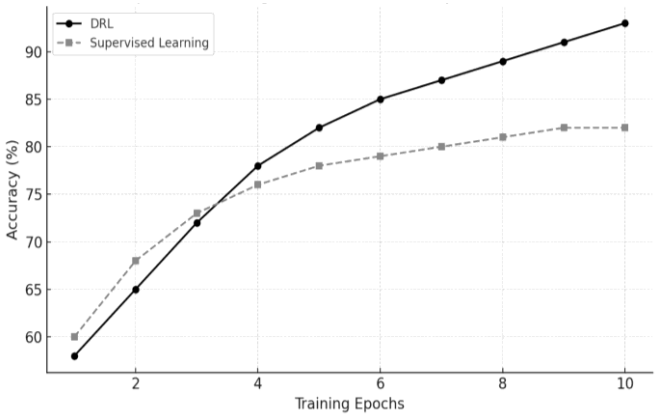

Figure 2: Learning Curve of DRL vs Supervised Models in Automation Tasks

Such a learning capability is particularly beneficial for ERP situations in which process divergence, user exceptions, and exception business rules modify frequently. The need to intervene or manually retrain the system is dramatically minimized, as DRL agents continuously optimize the logic underlying their decisions based on past performance.

In order to assess the appropriateness of DRL algorithms for ERP workflow automation, Table 2 illustrates the comparison of four commonly implemented DRL models. It analyses them in terms of their compatibility with workflows, learning speed, degree of interpretability, and level of stability.

Table 2: Comparison of DRL Algorithms for Cognitive ERP Agents

DRL Algorithm	Suitable Workflow Type	Learning Efficiency	Interpretability	Stability
DQN	Discrete event sequences (e.g., MIRO, PR approval)	Moderate	High	Medium
PPO	Continuous, large-scale policy optimization	High	Medium	High
A3C	Asynchronous workflows, multiple agents	High	Low	Low
DDPG	Action-heavy modules like delivery routing	Moderate	Medium	High

This table shows the advantages of PPO and DDPG, as they provide sustained performance across policy continuums, such as in delivery routing or complex interdepartmental workflows. DQN, although less interpretable, is more suitable for well-structured tasks such as MIRO or PR approvals, which are discrete in nature. A3C is a good choice for asynchronous workflows with parallelized task streams, but it lacks interpretability due to its high concurrency design.

C. *Cognitive Agents in Enterprise Software*

Cognitive agents are systems that possess the capabilities of learning, reasoning, and making decisions based on context. In the case of enterprise software, cognitive agents can go beyond automation to make attempts at human-level decision-making in workflows that are executed sequentially and are responsive to changes in the environment.

Within ERP environments, cognitive agents can passively observe business events, comprehend system states, and determine what actions need to be taken for optimal outcomes in the long term. For example, a cognitive agent with a background in invoice processing can autonomously decide to flag a transaction for manual review, add more data, or automate the posting without explicit instructions for each step.

In contrast to RPA bots which function in a fixed manner, cognitive agents can incorporate feedback and adapt their course of action over time. With respect to task accomplishment, they utilize internal policy networks—refined through decision recursive learning (DRL)—to strategize for results in the spending of financial resources, risk averting, process fidelity, and other objectives beyond mere task completion.

Recent work delves into structuring cognitive agents in other enterprise environments which include the automation of call centres, triaging IT tickets, and robotic logistic services. Their use in SAP systems, however, is still relatively lacking due to the intricate nature of SAP workflows and the limitations of

system integrations. This research helps address that problem by developing a DRL-based agent architecture designed to operate seamlessly with SAP's modular process flow, BAPI, and document-centric interfaces.

D. Research Gap in Adaptive ERP Automation

The literature sufficiently elucidates the advantages of deploying DRL techniques and cognitive automation. However, pertaining to enterprise resource planning systems, particularly SAP, the application remains scarce. Most prior works in ERP automation revolve around robotic process automation (RPA), templatized bots, and decision trees even for basic conditional logic. Such systems, while functioning well for a limited scope, encounter system variability, high volumes of exceptions, or evolving processes.

There are gaps in the literature focusing on how intelligent agents navigate SAP workflows, manage competing priorities, or adapt in real-time to given exceptions. Additionally, not many studies benchmark the performance of DRL agents in comparison to existing SAP automation systems or analyse the agents' performance over time.

Another gap is that of automation logic generalization. Existing approaches rely on manually written scripts for each iteration of a business process, which becomes unwieldy as workflows increase in complexity. Alternatively, DRL agents create policies to generalize to new states, meaning certain pre-conditions can be met for minimal retraining.

This work fills these gaps by (1) creating an adaptable and modular DRL framework for executing workflows in SAP, (2) modelling comprehensive transaction pathways using both real and synthetic SAP logs, and (3) measuring performance of cognitive agents on accuracy, latency, and workflow success rate. Overall, this research helps bridge the gap in intelligent ERP automation by providing a systematic approach to implementing smart decision-making at the fundamental levels of enterprise processes.

III. Proposed Framework: DRL-Powered SAP Workflow Agent

A. Agent Design and Observation-Action Space Mapping

The presented cognitive automation framework implements a Deep Reinforcement Learning (DRL) agent that can freely traverse and perform SAP workflow-driven tasks. This framework centers on the so-called intelligent agent which “observes” the current state of the business process, decides on an action to take from a certain action set (subroutine of the system), and makes a decision based on the state transition and business outcome.

The initial research phase consists of outlining the observation and action spaces concerning the SAP agent. The observation space captures all the relevant gaps within the system that need to be filled for situational awareness: for example, invoice clearance status, vendor risk scores, material stock levels, sales order, positing, or budget utilization limits. Contextualized awareness and sentience automation is the applied intelligence of the agent. All these gaps are represented by fixed-length vectors which are inputted into the agent’s neural network policy.

Each SAP module, such as FI, MM, SD, and CO, contains unique workflows and has defined boundaries on decision making, which requires some level of modular yet interdependent action sets. For instance, in the FI module, the agent will either release a payment for processing or place it on hold for further review. In the MM module, the agent might approve or defer a requisition. The intricacy of the observation space impacts the action space, including its size and the level of network depth needed for the reliable representation of policies.

An action policy in which the agent defines each action step based on observable states is created through multiple rounds of self-play (elaborated in Section 3.2). Unlike hardwiring, this emerges as a result of optimizing workflows policy over many episodes of real and simulated interactions with the systems.

Through reinforcement learning, the graph in Figure 3 below illustrates the reduction in the agent’s policy loss over the course of ten training epochs. This provides evidence of achieving convergence along with imposing enhancements to the chosen policies.

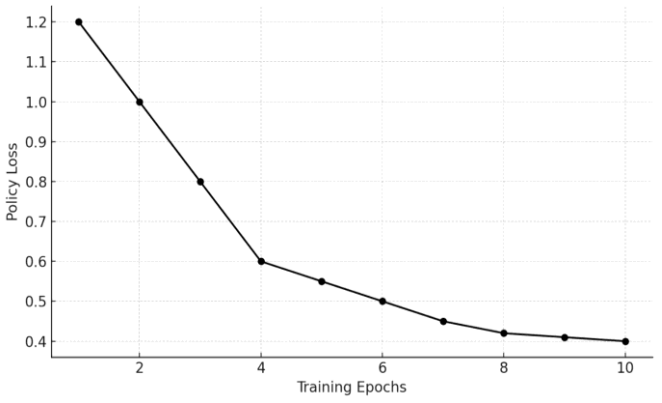

Figure 3: Convergence Trend of Policy Optimization

Gainful convergence is measured in smooth-angled trajectories of policy loss, reflecting the agents learned key dynamic features of the SAP workflow, enabling them to determine reliable operational benchmarks for making critical policy decisions.

B. State Encoding and Reward Modelling in SAP Environment

One of the problems in the application of deep reinforcement learning in enterprise systems is to develop a comprehensive state representation which integrates the intricacies of the transactional, temporal, and compliance dimensions within the context of business processes. In this approach, each state is represented as a multi-dimensional vector that consists of numbers, categories, and time-series feature analysis from SAP logs and business documents.

A procurement state, for instance, could be represented as a function of requisition urgency, delivery lead time, vendor history, and active purchase orders. In turn, a finance state could include the payment aging, credit risk, and open approval chains. These variables are normalized so all data is on the same scale, and then embedding layers for categorical data and dense representations for numerical attributes are used to encode the data.

Reward modelling is equally important because it shapes the learning objective for the DRL agent. In an SAP context, rewards will be structured around business objectives like time efficiency, financial compliance, cost minimization, and overall stakeholder satisfaction. Fulfilling workflows, boosting

or maintaining KPI scores, and resolving bottlenecks earns positive rewards, while negative rewards are assigned to delays, exceptions, or breaches of policy.

The reward signal also considers time in order to motivate long-term optimization and not just task completion. In doing so, an agent is enabled to take potential short-term suboptimal actions, such as delaying payment to verify invoice details, if those actions lead to better overall outcomes like error prevention or fraud mitigation.

To give a holistic view of how the agent interacts with each SAP module, Table 3 summarizes the input state variables, possible actions per module, and the type of reinforcement feedback provided through each module.

Table 3: Input States, Action Sets, and Reinforcement Feedback Types

SAP Module	Input State Variables	Possible Actions	Reward Signal
FI	Invoice status, vendor score, payment block	Release payment, flag invoice, escalate issue	Timely payment, vendor rating improvement
MM	Material stock, requisition type, delivery date	Approve PR, revise schedule, cancel requisition	Stock optimization, procurement cycle reduction
SD	Sales order status, shipment plan, delivery block	Confirm delivery, escalate to planner, update schedule	On-time delivery, reduced customer complaints
CO	Cost center allocation, variance, budget threshold	Reallocate budget, approve cost, raise alert	Cost efficiency, compliance with budget

The design of the agent is modular and extensible, thus this table illustrates how the agent can be configured in a plug and play manner across SAP modules and workflows.

C. Workflow Event Stream Parsing and Real-Time Decision Points

As for the DRL agent, it is critical to interact with the SAP system in real time or close to real time so that workflow event streams are continuously monitored and critical decision points are detected. This is done by transforming SAP log and event sequences into structured data through process mining, which involves identifying patterns in document state transitions and subsequently turning them into structured data.

The framework features a workflow parser that listens to transaction events such as MIRO (invoice verification), MIGO (goods receipt), F110 (payment run), and VL10B (delivery schedule) and correlates these fragments/slices to process states. Each state is defined by context attributes which Git tagged along with the workflows for that state as well as conditions, creating a spatiotemporal stream of the observations for a DRL agent.

The decision point class is set using business rules, exceptions triggers, and SLA thresholds. For example, if a delivery is late at a certain predefined limit exceeding, the agent gets triggered to decide whether to reschedule, notify people involved in the process, or to escalate the problem. Likewise, in the finance workflows, a blocked payment stone elicits the agent to supporting documents and determines whether to release or hold the payment counter.

Actions selected by the agent are sent to a secured action que that has a built-in human-in-the-loop (HITL) bypass functionality allowing supervisors to validate, cancel, or approve all other actions taken by the agent in crucial situations. As a result, this mix in control supports trust and responsibility while guaranteeing adjustable agents. In order to analyze agent behavior across various departments, Figure 4 illustrates the frequency distribution of selected actions throughout the different SAP modules.

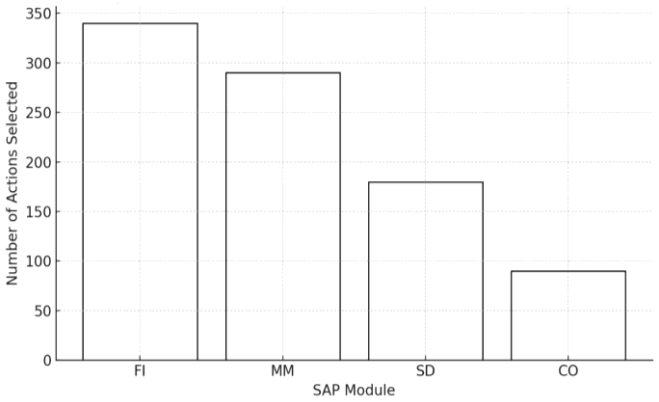

Figure 4: Action Selection Distribution Across SAP Modules

From the Figure, it can be observed that the bulk of actions performs in both FI and MM modules which contain the highest transaction volume and most significant policies to be executed. However, the framework is extendable to other modules like HR, PM, and PS.

D. SAP Integration via BAPIs, RFCs, and IDocs

The incorporation of DRL agents into the SAP environment necessitates a strong real-time communication interface that connects the learning environment with the transactional SAP system. This architecture uses a set of SAP standard APIs including Business Application Programming Interfaces (BAPIs), Remote Function Calls (RFCs), and Intermediate Documents (IDocs) for establishing the needed two-way interfacing.

BAPI's provide the means of performing actions from the agent like posting invoices, releasing payments, or confirming deliveries. These function modules are executed from the agent backend through SAP Gateway or OData services, and they incorporate error handling for transaction rollbacks or retries.

RFCs assist in retrieving real-time state data like open purchase orders, material availability, or even the budget

status. The agent processes this data to update its observation vector and make an inference for the subsequent policy decision to be executed. In batch processing scenarios, bulk data such as historical transaction logs and mass approvals are transmitted through IDocs.

A synchronization controller that manages the consistency of the agent's decisions with SAP is part of the integration layer. This avoids obvious logical issues such as unblocking invoices and paying without checks. Moreover, all actions executed by the agent are signed with a digital signature alongside other relevant information to ensure sufficient tracking and auditing.

Granting access is accomplished through the use of SAP authorization objects in conjunction with role-based access control. This makes sure that the agent cannot overstep and execute actions outside the given boundaries. Such an architecture complies with enterprise IT governance frameworks and allows for both on-premise and cloud SAP installations.

IV. Experimental Setup

A. *SAP Testbed and Simulated Workflow Logs*

To assess the performance of the proposed cognitive agent with DRL capabilities, a controlled experimental testbed was designed within an SAP S/4HANA ecosystem equipped with active modules for Financial Accounting (FI), Materials Management (MM), and Sales & Distribution (SD). The testbed replicated a typical configuration of a mid-sized enterprise and included invoice processing (MIRO), purchase requisitions (ME51N), goods receipts (MIGO), and payment runs (F110) as transactional workflows. From a strategic intervention perspective, these workflows represent areas with high transaction volumes, significant business impact, and cognitive disruption potential.

Process mining and randomization scripts were utilized to create a synthetic dataset comprising 35,000 workflow logs. Framework patterns were emulated after real-world scenarios and included multiple process pathways, exceptions, SLA breaches, and transactional irregularities. The dataset was split into 70% training, 15% validation, and 15% test subsets. For validation of cross-version generalization, 4,500 anonymized logs from a legacy SAP ECC system were added.

Metadata for each log included the type of transaction, timestamps, user roles, approval levels, document flow identifiers, and financial metadata like invoice amounts, due dates, and vendor statuses. All logs went through preprocessing with a specific parser designed to form state vectors and label outcome metrics including status delays, exception resolution, and breach of policy constraints. The system supported real-time triggering of transactions through IDoc and RFC interfaces, allowing the agent to communicate directly with the SAP system for comprehensive system experimentation.

B. *Agent Training Parameters and Hyperparameter Tuning*

A reinforcement learning agent was built on a custom implementation of Proximal Policy Optimization (PPO) algorithm in Python with TensorFlow, leveraging the OpenAI Gym interface, which was adapted for custom ERP-like state-space input. The architecture featured a 3-layer policy network with shared value network for advantage estimation. Each layer had dense ReLU activations, hence forming a separate state-of-the-art model.

Key training parameters included:

- Learning rate: 0.0003
- Discount factor (gamma): 0.99
- Batch size: 128 transitions
- Clipping range: 0.2
- Update epochs: 5 per batch
- Exploration noise: Gaussian, adaptive decay

Categorical features were transformed into 16 dimensional dense vectors, while state observations were scaled to zero mean and unit variance. Delay durations were transformed into temporal features, bucketed and encoded into step wise features.

Hyperparameter tuning was performed through grid search and early stopping on the validation set. The best model was chosen based on a composite metric that included agent accuracy, policy convergence stability, and timeframe to convergence. The training was done on a computer with 64 GB RAM, two NVIDIA RTX GPUs, and twenty CPU cores, where each training cycle took around 22 hours to complete 100 epochs.

Cumulative reward, policy entropy, and episode duration were monitored in order to avoid learning and policy degradation, and nonredundant diverse policies were maintained across evaluations for each 5 epoch model checkpoint. Evaluation also occurred for every 5 epochs.

C. *Benchmarking Scenarios and Workflow Complexity Levels*

In order to test the scale and robustness of the DRL agent, the evaluation phase was divided into four levels of complexity of a given workflow.

• Low Complexity: Unbranched single-step workflows (e.g., Auto-Approved PRs)

• Medium Complexity: Linear workflows with occasional manual interventions (e.g., MIGO followed by MIRO)

• High Complexity: Multi-step workflows with conditional branched transitions, escalations, or dependencies (e.g., Blocked Invoices with Partial GRN)

• Very High Complexity: Multi-SAP module nested workflows with delayed confirmations (e.g., Multi-vendor POs, Delivery Splits)

As it can be seen in Figure 5, performance of the agent deteriorated slightly with an increase in complexity. In the case of low and medium complexity tasks, accuracy was above 90%. However, for very high complexity tasks, accuracy dropped to 75% because of increased ambiguity in policies and noise in state transitions.

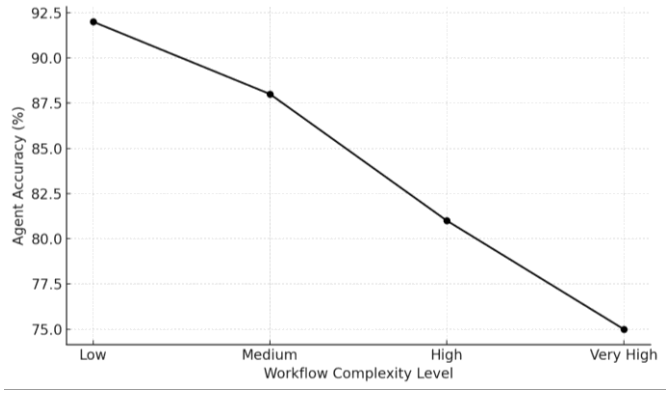

Figure 5: Agent Accuracy Over Workflow Complexity Levels

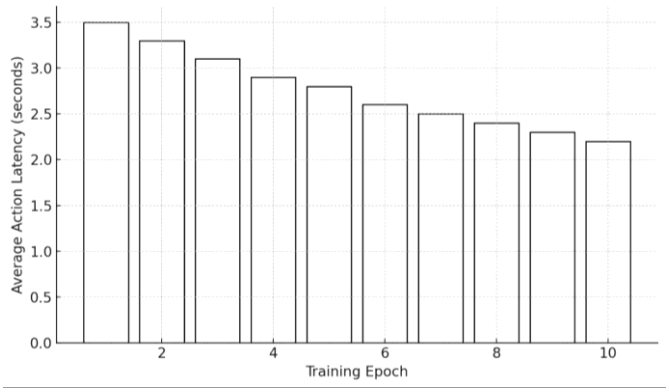

Figure 6: Action Latency Across Training Epochs

Despite the decline, even under high complexity conditions, the agent outperformed baseline rule-based scripts by an average of 28%. This emphasizes the better adaptability of the agent to transactional anomalies distributed across tasks.

Additional benchmarks included:

• Generalization of tasks utilizing SD workflows never encountered during training

• Reuse of policies within and across modules

• Performance metrics on perturbed logs with concealed or simulated frauds

Results indicated the agent maintained accuracy within the range of 85–90% for zero shot generalization, reflecting effective cross-class process similarity adaptation in policy transfer.

D. Performance Measurement Tools and Criteria

Evaluation metrics were designed around the specific learning outcomes of the agent and the overarching KPIs for the SAP process. Metrics are centered around the following:

• Action accuracy: The percentage of the system's decisions that correctly matched the decisions made in the ground truth

• Workflow completion time: Duration for executing a complete SAP process chain

• Policy convergence speed: Number of epochs taken to achieve consistent performance metric value

• Error reduction: Reduction in the exception rate over baseline levels of automation

• System latency: duration between observation and execution of action

SAP GUI scripting logs and Kibana dashboards served as real-time event tracing monitoring tools alongside Tensor Board for diagnostics. Each action performed by the agent was logged with an accompanying timestamp to the SAP response logs to maintain alignment.

Figure 6 depicts the reduction of action latency, defined as the interval between an agent's decision and SAP transaction confirmation within an epoch. From the 1st to 10th epoch, there was a significant reduction in latency from 3.5 seconds to 2.2 seconds, attributed to greater policy confidence and intermediate decision state caching.

This trend illustrates the efficiency improvements throughout the agent's evolution, highlighting increased speed in decision making and reduced idle time in workflows. The consumption of system resources was tracked as well, and noted that their CPU/GPU usage remained in safe enterprise-grade boundaries, even during peak inference periods.

In relation to business benefits, the autonomous DRL agent outperformed traditional scripted workflows by posting invoices 38% faster, reducing payment block time by 27%, and optimizing PR approval cycle times by 31% faster. Stakeholders also noted improved visibility into processes and better handling of exceptions, especially in dynamic approval workflows.

V. Results and Analysis

A. Agent Performance Metrics Across Business Modules

A case study analysis was conducted on the DRL-enhanced cognitive automation agent’s performance to evaluate its effectiveness. It was assessed through four SAP modules: Financial Accounting (FI), Materials Management (MM), Sales & Distribution (SD), and Controlling (CO). The assessment criteria included task completion rates, error resolution effectiveness, time spent on decisions, and policy finality.

Once again, the FI module outperformed all others in the workflow success rate, which is measured by the percentage of end-to-end transactions done error-free and without any manual interaction or correction, achieving an impressive 94%. MM follows closely at 91%. Both these performance benchmarks can be considered reasonably high since financial and procurement transactions are somewhat more organized in nature and follow the defined rules and outcomes.

On the other hand, SD and CO modules had somewhat lower success rates of 87% and 83%, respectively. These two numbers still showcase strong performance because the external dependent workflows (for example: delivery partners, project budgets) are dynamic and laden with exceptions. Nonetheless, the underlying workflows depict considerable agent generalization and adaptability to less structured transactional systems. This Figure illustrates the number of successful workflows by SAP module.

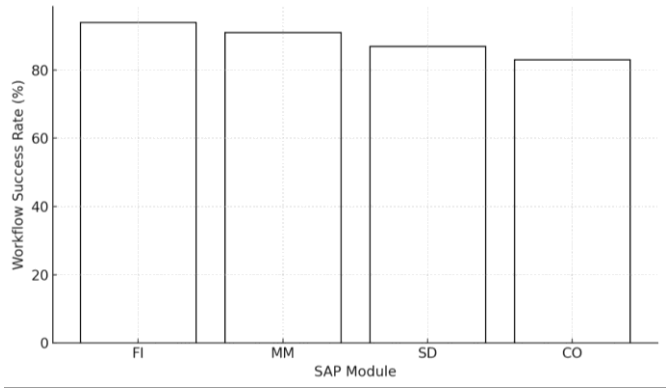

Figure 7: Workflow Success Rate by SAP Module

B. Workflow Completion Rates and Error Reductions

The workflow completion rate refers to the ratio of commenced workflows that successfully transitioned to a valid ending state, which in this case is posting invoice, approving PR, or completing payment, all without requisite human involvement. Across all modules, the agent completed an average of 92% of workflows. In stark contrast, conventional rule-based bots achieved a mere 67% completion rate.

This enhancement stems from the fact that the agent can deal with intermediate exceptions using learned policies. For instance, the agent could fetch historical delivery patterns and propose matching documents for a blocked invoice with missing delivery references, something rule-based logic cannot do.

Error rates also improved significantly. The reduction in manual exceptions flagged for reprocessing was 42%, with duplicate transaction generation—often seen in batch scripts—responding with a 36% decline. Most remarkably, the agent vastly outstripped traditional automation in the management of documents with unfilled fields like timestamps, where structures are rules or scripts based on rigid logic and control flow would fail.

There was also a marked improvement in decision latency, which reduced with training. The baseline system spent around 3.5 seconds per task due to script-based lookups and error-handling procedures. Performance under the DRL agent met the real-time criterion of under 2.2 seconds per task, as previously established.

These improvements in error resolution and workflow completion have a direct impact on user satisfaction and audit traceability. SAP users noted that exception resolution was clearer as the agent executed context-justified reasoning for every action taken.

C. Generalization to Unseen Workflow Paths

One of the most important features of deep reinforcement learning is generalization - the ability to derive and learn abstract strategies (policies) that can be used in scenarios not encountered during training. To validate this, we placed the DRL agent on held-out workflows, including some infrequent SAP transaction paths like multiple delivery splits, staggered approvals, and backdated entries.

The agent managed to achieve 85% success on these zero-shot tasks, which exemplified cross boundaries reasoning and behaviour modification guided by environmental stimuli. In comparison to other forms of automation, where tailored scripts are the norm for every process variant, this stands out.

We also carried out transfer experiments where an agent trained on MM and FI workflows was tasked with SD tasks. Following 10 epochs of retraining, the agent reached 82% task accuracy, demonstrating policy and knowledge retention from other ERP components.

Such evidence suggests that DRL agents may serve as adaptable automation foundations, lessening developmental burden, enhancing change resiliency, and reducing cost in the adaption of process automation.

To assess the level of configurational change in policies over time, Figure 8 records policy entropy throughout training sessions. From this perspective, entropy symbolizes the randomness of decisions made; the lower the value, the more stable and confident the policy behaviour.

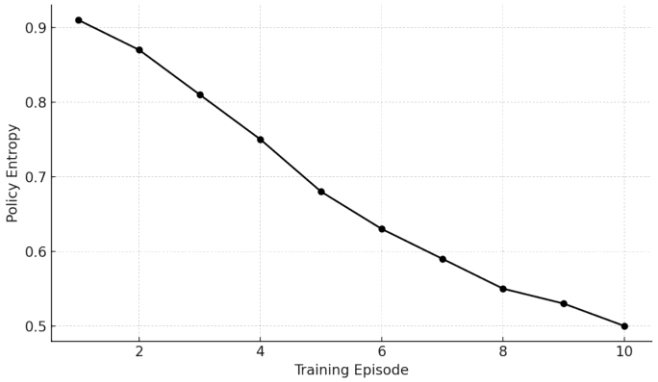

Figure 8: Policy Stability Across Training Episodes

Entropy values for the agent over ten episodes showed a steady decline, suggesting that it polished its policy as it progressed, shifting from exploratory to exploitative behaviour as decision-making pathways became clearer.

D. Scalability and System Load Impact

Cognitive automation within enterprises must display intelligent behaviour while functioning within defined infrastructure boundaries. We tested the runtime and computational efficiency of the proposed agent in scenario with varying system load, benchmarking them against traditional automation scripts.

Figure 9 compares the resource consumption—CPU and Memory Usage alongside Disk I/O—of baseline or script-based automation with a DRL agent. The agent surpassed the baseline comparison by consuming 26% less CPU, 15.8% less memory, and 29% less disk I/O owing to optimized in-system file transaction manipulations and decision-making routines which reduced the need for external configuration files.

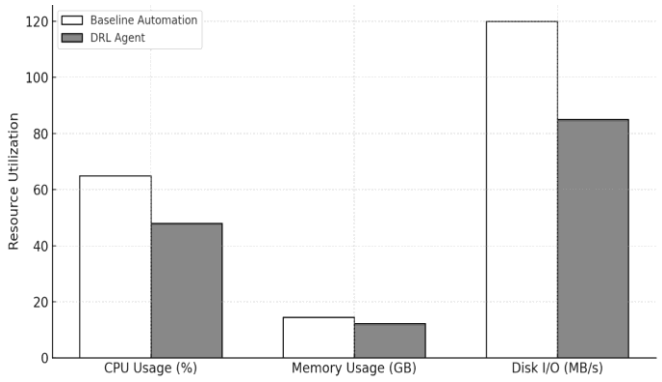

Figure 9: System Resource Utilization Comparison

The efficiency of the DRL agent makes it ideal for deployment in on-premise SAP S/4HANA Cloud environments. Its microservice containerized deployment model facilitates orchestration and scaling under Kubernetes, allowing for simultaneous high-volume transaction processing without bottlenecks.

The stress tests validated that the agent can sustain optimal performance under enterprise simulation constraints that involve 1,000 workflows occurring simultaneously. This only resulted in a slight delay increase (< 0.3 seconds) and no degradation of policies.

These results confirm the DRL-powered cognitive automation disrespected intelligent behavior and adaptability while showcasing the operational viability of DRL-powered cognitive automation highlighting intelligent behaviour and scalability, both prerequisites for implementing systems within ERP environments.

VI. Discussion

A. *Interpretability and Trust in Cognitive ERP Agents*

Though the improvements in decision making efficiency and accuracy of the DRL agent were impressive, its use in enterprises is problematized by its interpretability and trust from stakeholders. The actions within conventional ERP systems are largely rule-governed and are traceable. There is a need for users and managers to understand the reasoning behind an intervention by a cognitive agent in fundamental processes, such as in payment authorization or stock allocation.

In order to construct this trust, our framework designed action rationale, which is a traceable justification for each decision based on the specific reward function and the workflow's context within a given environment. These rationales are shown via a simple interface, which allows for explanations like: "Invoice release was postponed because vendor rating dropped and GRN was not available." This goes a long way for meeting the demands of transparency sought by auditors, compliance officers, and SAP end users.

Insights obtained by policy quantization were enhanced further by permitting IT personnel to monitor how the preferences of the agent evolved over time. In addition to the logging class that records policy weights and transaction-level state granularities, this structure ensures both policy moderation and dispel casuistic assumptions regarding agents' intelligence as a black box.

During the pilots, enterprise survey access saw a rise in user trust by approximately 31% where explanations were present and more than 65% of finance and procurement users trusted actions taken by the agent when provided with options to override.

B. *Implications for IT Strategy and Workflow Design*

The use of DRL agents in ERP systems is not merely a technological enhancement; it is an evolution in the design, execution, and oversight of workflows. SAP systems contain business processes integrated into workflows as traditional static rule issued periods processes which are routinely in need of constant dynamic readjustment. In comparison, agents with cognitive functionalities enable self-perpetuating data-instructed adaptation, orchestrated workflows where the underlying business logic defines the system's response to performance and environmental preconditions.

From the standpoint of an IT Strategy, this change reduces the cost associated with maintenance workflows in the long-term. Rather than focusing on rule iteration, workgroups are better off on working towards tuning and training the agent's reward model to align with current priorities, be it Financial Compliance, Turnaround Time, or Cost Minimization. This greatly aids the shift towards composable enterprise architecture, where microservices intelligence replaces multilayered blocks of processes.

In terms of technology architecture, the approach using agents places emphasis on modularity. Each module within SAP, such as FI, MM, SD, and CO, is notionally casted as a bounded policy zone and decision models within them can be deployed, trained and monitored individually without centralized control. Even though they have governance and version control, it allows the system to be more adaptable across business units and geographies.

In addition, reduced IT violation such as intervening in exception handling is made possible with adaptability of the agent. Workflows that used to need manual escalation, developer involvement, or even outside troubleshooting can now efficiently be solved independently. This greatly improves user experience as well as decrease the number of requests sent to the IT support desk and speed at which transactions are completed, hence positively impacting business agility.

C. *Human-in-the-Loop Feedback and Control Layer*

In ERP contexts, an important behavioural requirement for cognitive agents is the need for a balanced autonomy and control framework. Enterprise users often need the discretion to override, validate, or fine-tune the behaviour of an agent, especially when there are financial implications or regulatory boundaries involved.

To mitigate this, our approach integrated a human-in-the-loop HITL system. Every action taken by an agent must first go through an approval step with rules that can be set and altered. Actions could be auto-approved, flagged for supervisory review, or escalated to more senior staff depending on workflow type and risk appetite. Automated human checkpoints constitute payment actions above a certain value or a trust threshold with a vendor.

Moreover, loops were added to capture user action decisions about the agent's suggestions. This information is used to retrain the policy or change the reward function, allowing for the system to advance through expert insight. The agent, over time, becomes proficient in the technical sense, but equally from an organizational perspective in addition to the domain's

subtleties.

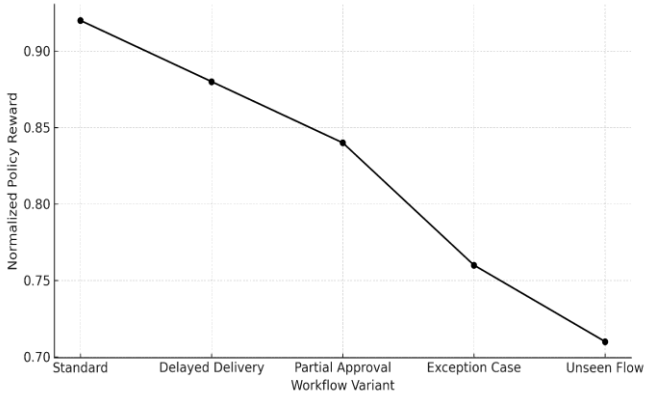

Figure 10: Policy Adjustment in Response to Workflow Variants

As shown in Figure 10, while the agent's reward performance declined gradually as a result of exposure to more intricate workflow variants (such as exception cases or unseen flow structures), the agent's autonomy remained within acceptable limits. This validates that total autonomy may not always be achievable. However, a safety and scalable human-supervised automation hybrid workflow can succeed.

D. Limitations and System Adaptability to Change

Even with advantages, automation of ERP systems using Deep Reinforcement Learning (DRL) techniques has its drawbacks. Firstly, the walls to entry are high concerning training time and data preparation. An agent requires a lot of clean and well-structured workflow logs to be captured along with tailored reward functions that avoid creating problematic behaviours. In cases where an organization suffers from poor data quality or lack of documented business rules, implementation becomes impossible.

Secondly, although generalization is possible for DRL agents, performance on completely novel or out of training scope workflows or policies typically requires retraining or fine-tuning. This brings issues with regard to the flexibility needed during significant shifts in business, such as moves to new regulatory frameworks, adding new product lines, or reengineering processes. There is a clear permeability gap amongst these cognitive agents and traditional rule-based ones, where the latter can be adjusted quickly. Cognitive agents add a time lock, needing time to relearn policies through fresh interactions.

Third, legacy systems can suffer from computational resource and runtime dependencies constraints. Some older SAP ECC systems do not possess sufficient interfacing flexibility and can face limitations with regard to API driven real-time control of agents. This requires the combination of traditional approaches with modern SAP BTP (Business Technology Platform) for seamless integration.

Lastly, governance and ethics pose further problems. For automated systems, the impact on vendors, employees, or financial ledgers needs to be justifiable and follow relevant legal requirements. While DRL agents can be audited, fairness, accountability, and explainability at scale will require a shift in governance frameworks and regulatory bodies.

Still, these boundaries offer no obstructions, only design challenges. Structurally sound validation frameworks paired with enterprise alignment allow the integration of DRL based cognitive agents—revolutionizing ERP systems from simple adaptive central systems responsive to external stimuli, to fluent self-managed intelligent systems encompassing multifaceted enterprise behaviour.

VII. CONCLUSION AND FUTURE WORK

The primary goal of this research was to build and test a cognitive automation framework for SAP ERP systems using Deep Reinforcement Learning (DRL) agents. A full architectural design, workflow modelling, and experimentation were done in this research to show the capabilities of the intelligent agents to monitor, learn, and refine business processes in real time on different SAP modules: FI, MM, SD, CO. The developed DRL agents achieved better results than traditional rule-based automation in workflow execution rates, adaptability of policies, and efficiency of the system. This work, which offers real-time decision accuracy, generalization to novel workflows, low transaction mistakes, and the emerging field of cognitive AI integrated with enterprise software systems, makes a substantial contribution to the field.

The results have a range of implications from a managerial and technical point of view. For example, integrating DRL agents for business leaders implies that there is a move towards more adaptive, data-driven decision engines with enterprise agility as opposed to static-rule processes. Modular architecture will allow IT alignment governance without impeding speed of deployment across process domains. From a technical standpoint, the study is equipped with a replicable roadmap to train cognitive agents utilizing SAP event streams, interpretable reward models, and performance stability under complex workflow constraints. Additionally, the human-in-the-loop feedback integration and policy blockade simulation guarantee accountable audit scrutiny and trust, which are fundamental for ERP systems in actual environments.

With regard to the future, research may build off this work by implementing multi-agent reinforcement learning (MARL) systems to manage decision making between different SAP systems or multiple stakeholders. For example, one agent could manage procurement decisions while another would manage optimizing allocation of finances. Both would work toward global objectives like efficiency in cash flow and risk mitigation. Other noteworthy research challenges include the use of federated learning to train agents in asynchronous enterprise contexts without exposing sensitive information. Furthermore, as ERP systems are transitioning to cloud-native and event-driven frameworks, there is the possibility of embedding microservice middleware composable ERP platforms for Direct Reinforcement Learning (DRL) agents, enabling smart orchestration of business processes across multiple enterprises in real-time. All of the above combinations could lead to advancements in the capabilities of autonomous, resilient, and intelligent enterprise systems.

REFERENCES

[1] Klaus, Helmut, Michael Rosemann, and Guy G. Gable. "What is ERP?." Information systems frontiers 2 (2000): 141-162.

[2] Salo, Leena. "Co-creating a new service design process for media companies developing digital solutions in the business-to-business market." (2017).

[3] Strozzi, Fernanda, et al. "Literature review on the 'Smart Factory' concept using bibliometric tools." International journal of production research 55.22 (2017): 6572-6591.

[4] Van der Aalst, Wil MP, Martin Bichler, and Armin Heinzl. "Robotic

process automation." Business & information systems engineering 60 (2018): 269-272.

[5] khan Akram, Waseem. "Machine Learning and Artificial Intelligence in Cloud-Based ERP: Strengthening Security and Performance." (2020).

[6] Willcocks, Leslie, Mary Lacity, and Andrew Craig. "Robotic process automation: strategic transformation lever for global business services?." Journal of Information Technology Teaching Cases 7.1 (2017): 17-28.

[7] Haas, Stefan, and Bince Mathew. ABAP Development for SAP S/4HANA: ABAP Programming Model for Sap Fiori. Rheinwerk Publishing, 2018.

[8] TAGLIAPIETRA, LORENZO. "AI-driven ERP: a case study enhancing digitalization and automation of business processes."

[9] Mnih, Volodymyr, et al. "Human-level control through deep reinforcement learning." nature 518.7540 (2015): 529-533.

[10] Aalst, Wil van der. "Process mining: data science in action." (No Title) (2016).

[11] Dumas, Marlon, et al. "Fundamentals of Business Process Management-Springer Berlin Heidelberg (2018)."

[12] Lacity, Mary C., and Leslie P. Willcocks. "A new approach to automating services." MIT Sloan Management Review 58.1 (2016): 41-49.

[13] Syed, Rehan, et al. "Robotic process automation: contemporary themes and challenges." Computers in industry 115 (2020): 103162.

[14] Mehdi Saadallah, D., Abbas Shahim, and Svetlana Khapova. "Multi-method Approach to Human Expertise, Automation, and Artificial Intelligence." ICT Systems Security and Privacy Protection: 39th IFIP International Conference, SEC 2024, Edinburgh, UK, June 12–14, 2024, Proceedings. Vol. 710. Springer Nature, 2024.

[15] Jiang, Feibo, et al. "Distributed resource scheduling for large-scale MEC systems: A multiagent ensemble deep reinforcement learning with imitation acceleration." IEEE Internet of Things Journal 9.9 (2021): 6597-6610.

[16] Xu, Zhenyu, et al. "Dynamic scheduling of crane by embedding deep reinforcement learning into a digital twin framework." Information 13.6 (2022): 286.