RCMOS – Multidisciplinary Scientific Journal The Knowledge.
ISSN: 2675-9128. São Paulo-SP.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

ORBITA: An automation recommendation framework based on Task Mining and Large Language Models.

ORBITA: An automation recommendation framework based on Task Mining and Large Language Models

Márcio da Silva Bastos - Lattes: https://lattes.cnpq.br/4180847255464415; UNIFACCAMP Campo Limpo Paulista, SP, Brazil

Summary

Digital transformation drives organizations to seek operational efficiency through process automation. Robotic Process Automation (RPA) has established itself as a solution for repetitive tasks, but it presents a critical bottleneck in the discovery and documentation phase, which consumes up to 40% of the total project time and depends on scarce human specialists. Task Mining allows the identification of work patterns from interaction logs with graphical interfaces, but translating these patterns into executable scripts remains a manual activity and subject to errors. Large Language Models (LLMs) offer code generation capabilities, but present risks of hallucinations when applied directly to critical systems. This article presents the

Framework ORBITA (Orchestrated Recommender Based on Intelligent Task Analysis), a six-layer architecture that integrates Task Mining and LLMs mediated by Retrieval-Augmented Generation (RAG) to recommend automations with sandbox validation and human governance.
The framework was empirically validated in four representative domains: interface automation (RPA Challenge), web data extraction (Books to Scrape), document processing (Invoice Generator), and data pipeline (Kaggle Datasets). The results indicated an average reduction of 97.3% in Mean Automation Time compared to manual development, with a 100% rate of functionally executable scripts after structural validation.
Keywords: Process automation; Task mining; Large language models; ORBITA framework; Robotic process automation; Retrieval-augmented generation.

Abstract

Digital transformation drives organizations to seek operational efficiency through process automation. Robotic Process Automation (RPA) has consolidated as a solution for repetitive tasks but presents a critical bottleneck in the discovery and documentation phase, consuming up to 40% of total project time and heavily relying on scarce human specialists. Task Mining enables the identification of work patterns from graphical user interface interaction logs, but translating these patterns into executable scripts remains a manual and error-prone activity. Large Language Models (LLMs) offer code generation capabilities but present hallucination risks when directly applied in critical systems. This paper presents the ORBITA Framework (Orchestrated Recommender Based on Intelligent Task Analysis), a six-layer architecture combining Task Mining and LLMs mediated by Retrieval-Augmented Generation (RAG) to recommend automations with sandbox validation and mandatory human governance. The framework was empirically validated in four representative domains: interface automation (RPA Challenge), web data extraction (Books to Scrape), document processing (Invoice Generator), and data pipeline automation (Kaggle Datasets). Results demonstrated an average 97.3% reduction in Mean Automation Time compared to manual development, with a 100% rate of functionally executable scripts after structural validation.

Keywords: Process automation; Task Mining; Large Language Models; ORBITA framework; Robotic Process Automation; Retrieval-Augmented Generation.

1. Introduction

Digital transformation has created increasing pressure on organizations to scale up their operations. Process automation initiatives. Robotic Process Automation (RPA) occupied a position

RCMOS – Multidisciplinary Scientific Journal The Knowledge.
ISSN: 2675-9128. São Paulo-SP.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

Central to this movement is enabling software robots to perform repetitive tasks.
by humans in graphical interfaces. However, the typical lifecycle of an RPA project presents
Recurring bottleneck: the discovery and documentation phase, which precedes the coding of bots, consumes
approximately 40% of the total project time and requires the presence of specialists with
technical and business knowledge (JIMENEZ-RAMIREZ et al., 2023).

Task mining emerged as a discipline to tackle this problem empirically. Instead
Instead of conducting interviews and direct observations, Task Mining systems automatically capture
Interface event logs are analyzed and machine learning is applied to identify recurring patterns.
of work (REHSE; HASSELBRING, 2021). The result of mining, however, remains
Syntactic: we know what the user did, but not why they did it, and even less how to transform that.
Knowledge of reliable executable code.

Large Language Models (LLMs) such as GPT-4 and Claude have brought a new perspective to...
demonstrate semantic comprehension skills and the ability to generate code of reasonable quality from
of textual descriptions (ROSSI et al., 2023). The direct application of LLMs to Task Mining data
However, it is problematic: hallucinations, inconsistencies between executions, and a lack of mechanisms.
Validation processes make these models unsafe for direct use in production environments (BERTI;
KOURANI; VAN DER AALST, 2024).

This article presents the ORBITA Framework (Orchestrated Recommender Based on
Intelligent Task Analysis), an architecture that connects these three technologies through mechanisms
risk mitigation. ORBITA is organized into six functional layers: local event capture,
pattern mining, RAG-based code generation, structural validation in a sandbox,
Versioned knowledge storage and human governance interface. The central hypothesis is
orchestrating these capabilities transforms passive discovery into actionable recommendations.
reducing the average automation time while maintaining adequate reliability for various environments.
regulated.

The work is structured as follows: section 2 presents the theoretical foundations.
and related works; section 3 describes the architecture of ORBITA; section 4 presents the study
from an empirical case study; section 5 discusses the results; and section 6 concludes with conclusions and perspectives.
of future work.

2. Theoretical foundations and related works

2.1 Robotic Process Automation and its bottlenecks

RPA enables software to operate graphical interfaces in the same way that humans do.

RCMOS - Multidisciplinary Scientific Journal The Knowledge.
ISSN: 2675-9128. São Paulo-SP.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

executing predefined sequences of actions on existing systems without the need for integration.
via API (JIMENEZ-RAMIREZ et al., 2023). The standard implementation cycle comprises five
Phases: candidate identification, expert analysis, requirements documentation,
Development and testing. Studies show that the first two phases consume approximately 40%.
of the total effort, and that RPA developers spend on average 60% of their time writing
Code that repeats patterns already implemented in previous automations (WANG et al., 2022). These
These characteristics created the research opportunity that ORBITA seeks to address.

2.2 Task Mining: Empirical Pattern Discovery

Task mining differs from traditional process mining in the granularity of the analysis. While
Process Mining operates on system transaction logs, while Task Mining captures events from...
The interface, including clicks, typing, and navigation, allows you to observe the actual work at a level of...
activity (REHSE; HASSELBRING, 2021). The main limitation of the discipline is that the standards
The identified elements are syntactic: they describe sequences of actions, but do not capture the intention or the
semantics of the process. The translation of these patterns into executable code still depends on
human intervention.

2.3 Large Language Models in process automation

LLMs such as GPT-4 demonstrate the ability to generate functional Python code from...
textual descriptions with a success rate of over 75% for well-specified problems (ROSSI
et al., 2023). Wang et al. (2024) extend these results to the domain of automation, translating
Process descriptions in executable specifications. Berti, Kourani and van der Aalst (2024)
They document that LLMs applied to Process Mining tasks exhibit consistent hallucinations.
Inconsistency between executions and vulnerability to syntactic variations in the input. These limitations
They require architectural mitigation measures before being used in production.

2.4 Retrieval-Augmented Generation

The RAG mechanism bases the generation of LLMs on examples retrieved from a database.
structured knowledge, reducing the dependence on the model's parametric knowledge and,
Consequently, the rate of hallucinations. Barbieri et al. (2025) report a reduction in hallucinations of
32% to 9% in Process Mining contexts when incorporating RAC. This evidence supported the
Architectural decision to integrate RAG as a central quality control mechanism in ORBITA.

RCMOS - Multidisciplinary Scientific Journal The Knowledge.
ISSN: 2675-9128. São Paulo-SP.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

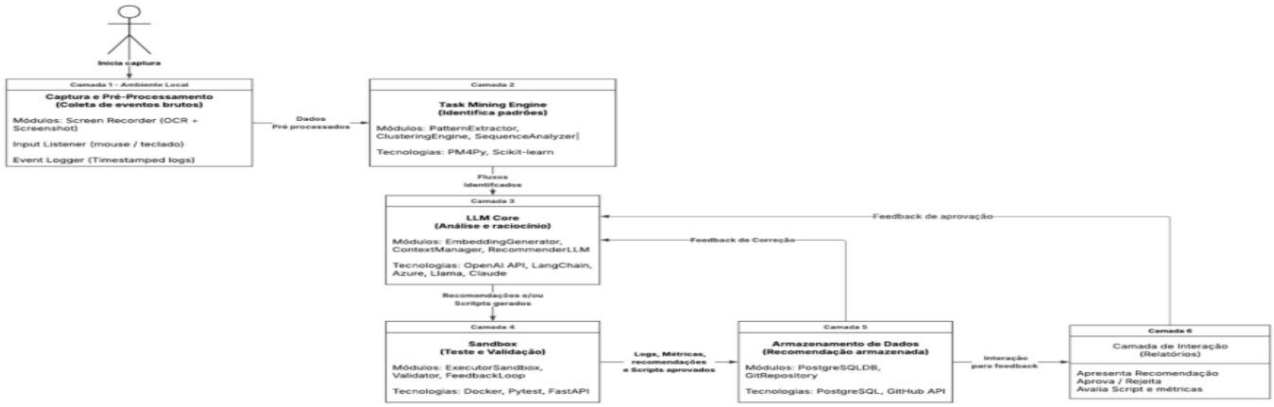
2.5 Related Works

Three lines of research directly precede ORBITA. The SmartRPA (AGOSTINELLI et (al., 2022) generates scripts from interface logs, but does not incorporate semantic interpretation nor Validation. PromptRPA (WANG et al., 2024) translates textual descriptions into RPA workflows. but it requires manual specification of the process, circumventing the problem of empirical discovery. The Log2Plan (LEE et al., 2025) integrates Task Mining and LLMs in a way that is closer to the proposal of ORBITA, however, does not perform sandbox validation and does not generate intermediate representations. structured process. ORBITA differentiates itself by systematically integrating five capabilities. identified as absent in the literature: automatic empirical discovery, semantic interpretation, Generation based on RAG, structural validation, and formal governance.

3. The ORBITA Framework

3.1 Architectural Overview

Figure 1 - Overview of the layered architecture of the ORBITA Framework



ORBITA is organized into six functional layers arranged in a sequential flow. A separation between Layer 1, which operates locally on the user's workstation, and the Layers Steps 2 through 6, which are performed in the central analytical core, address a recurring technical limitation: Web browsers operate in restrictive environments that prevent the capture of events in applications. Legacy desktop environments. The hybrid model adopted by ORBITA allows capturing events in diverse environments. heterogeneous systems such as browsers, spreadsheets, PDF viewers, and local ERPs in a way unified. Figure 1 presents an overview of the framework's layered architecture.

RCMOS – Multidisciplinary Scientific Journal The Knowledge.
ISSN: 2675-9128. São Paulo-SP.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

3.2 Layer 1: Orbit Discovery Recorder

The capture agent uses three simultaneous mechanisms: visual capture via the mss library. with configurable intervals, interception of mouse and keyboard events via Pynput, and extraction of Textual context via OCR with pytesseract on regions of interest. An activity classifier categorizes each event as PRODUCTIVE, NEUTRAL, or DISTRACTIVE based on lists.
Configurable. A privacy module automatically detects and redacts sensitive fields such as Passwords and personal documents are removed before export. The output is a ready-to-use structured JSON file. for ingestion by the analytical core.

3.3 Layer 2: Task Mining Engine

The MiningLayer module converts event sessions into compact text tokens and applies them. n-gram (1-2) vectorization with CountVectorizer. The KMeans algorithm partitions the sequences into k clusters, with k automatically determined by the Silhouette Score in the range of 2 to 6, avoiding dependency on manual parameterization. For each cluster, the canonical sequence is extracted. frequent, quality metrics such as intra-cluster cohesion and consistency, and n-grams Discriminative. The model is persisted to disk for reuse in subsequent sessions.

3.4 Layer 3: LLM Core with RAG

The intelligent core integrates Sentence Transformers for semantic embeddings and FAISS. for efficient vector search. The RAC mechanism retrieves the k most similar scripts from the database. knowledge is then injected into the prompt sent to the LLM, basing the generation on examples. Validated functional tests. The feasibility assessment considers the frequency of the pattern, consistency, and Cluster cohesion, producing three decision categories: AUTOMATION (score above 65 and cohesion above 0.65), IMPROVEMENT (intermediate score) and NO_AUTOMATION (score below (of 35). Each artifact includes executable code, suitability analysis, and traceability metadata. which document the RAG scripts used as a reference.

3.5 Layer 4: SandboxValidator

Structural validation of the generated code occurs before any presentation to the specialist. The SandboxValidator performs four checks: AST (Abstract Syntax Tree) parsing. for detecting syntax errors, analyzing security patterns to identify constructs.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

Critiques such as eval and os.system, mandatory structure checks such as proper imports to domain and exception handling blocks, and cyclomatic complexity calculation with alert in Values above 10. Scripts with critical errors are blocked; scripts with warnings progress with the documented reservations.

3.6 Layers 5 and 6: Persistence and Governance

Layer 5 stores approved scripts in a structured hierarchical JSON repository for future migration to PostgreSQL and integration with Git, while maintaining automatic versioning and Complete metadata. The FAISS index is updated with each approval, enriching the database. Recovery for future generations and implementing continuous organizational learning. The Layer Section 6 presents the complete recommendation to the specialist in five structured sections: decision of Feasibility study with score, semantic context retrieved via RAG, generated code, result of Validation in a sandbox environment and quality metrics of the identified standard. Approval or rejection requires... Structured justification, creating an auditable audit trail.

4. Empirical case study

4.1 Experimental design

The empirical validation of ORBITA was conducted through four independent scenarios. selected to maximize technological domain variability. The experimental protocol It followed three operational hypotheses: H2.1 (Technical Feasibility) predicted a functionally script rate. executables with a higher than 75% success rate in sandbox validation; H2.3 (Reduction of Average Time to Automation (TMA) anticipated a minimum 40% reduction in execution time compared to the process. manual; H2.2 (RAG Efficacy) predicted a reduction in hallucinations of at least 50% compared to generation without retrieval. For each scenario, the manual process was executed and timed as baseline before applying ORBITA.

4.2 Scenario 1: RPA Challenge

The RPA Challenge is a public benchmark for web form automation that changes dynamically changes the HTML attributes of the fields with each execution, preventing approaches based on Static identifiers. 147 mouse events, 1,847 keystrokes, and 273 screenshots were recorded. During capture, Task Mining identified a pattern of dynamic form filling with

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

cohesion of 0.78. LLM Core generated a Selenium script with retrieval of two similar examples via FAISS. The SandboxValidator approved the script in 283 ms. The script executed correctly 10 times. consecutive forms with 100% accuracy.

4.3 Scenario 2: Books to Scrape

The Books to Scrape environment provides a web catalog of books with a pagination structure. and variable attributes. The Orbita Discovery Recorder recorded 89 click events during the session. Navigation within the catalog. Task Mining identified a structured extraction pattern with a sequence. The code has a canonical structure of 6 actions and a cohesion of 0.81. The generated code used BeautifulSoup and pandas for extraction. HTML tabular. The execution extracted 30 books in 1.8 seconds with complete data integrity. verified.

4.4 Scenario 3: Invoice Generator

This scenario modeled the manual processing of invoices in PDF format with a record of fields extracted from a spreadsheet, a process that involves switching between PDF viewers, browser and Excel. 234 heterogeneous events were recorded covering the three applications, highlighting the Orbita Discovery Recorder's ability to capture events in multi-environments application. The generated code used PyPDF2 and openpyxl for batch processing. The SandboxValidator issued a warning about partial logging, but no critical errors. The execution Processed 20 invoices with an average processing time of 0.82 seconds per document.

4.5 Scenario 4: Kaggle Dataset

The fourth scenario covered a complete data acquisition and preparation pipeline starting from public repository. The manual process includes navigating to the dataset, downloading the ZIP file, Extraction, loading, and preliminary inspection of pandas. Task Mining identified a pipeline of The data has four stages: acquisition, inspection, cleaning, and export. The generated code used... Requests and pandas were used to reproduce the complete workflow. The execution produced a DataFrame with 8,807 records and 12 columns in 4 seconds.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

5. Results and discussion

5.1 Consolidation of results by scenario

Table 1 consolidates the quantitative metrics obtained in the four experimental scenarios.
Gathering capture data, execution times, and validation results in a sandbox.

Table 1 - Consolidation of quantitative results by experimental scenario

Scenario	Captured events	Manual time (s)	Automated time(s)	Reduction (%)	Sandbox Validation
RPA Challenge	147 clicks, 1,847 keystrokes, 273 screenshots	297	< 5	98.3%	Approved
Books to Scrape	89 click events	178	1.8	99.0%	Approved
Invoice Generator 234 events	heterogeneous (PDF) (Excel + browser)	185/doc	0.82/doc	99.6%	Approved with notice
Kaggle Dataset	Download and extraction events ZIP and handling CSV	312	4.0	98.7%	Approved

Source: Prepared by the author.

5.2 Status of the hypotheses

Hypothesis H2.1 was fully validated: all four generated scripts passed the test.
The structural validation of SandboxValidator was error-free, exhibiting cyclomatic complexity.
Between 2 and 4, a value associated with a reduced risk of defects. All scripts executed the task.
The intended result was successfully achieved in the target environment. The 100% rate exceeds the established target of 75%.
hypothetically.

Hypothesis H2.3 was also validated by a significant margin. The average reduction in time
The task completion rate was 97.3%, placing it between the 98.3% of Scenario 1 and the 99.6% of Scenario 2.
3. This magnitude should be interpreted with caution: the time measured refers to the execution of the script.
generated, not the complete cycle of capture, processing, generation, and approval. In a deployment
With mandatory human supervision, the end-to-end time would be longer. Still, even
Considering the capture and processing time, the advantage over development
traditional manual, which requires between four and twelve weeks for automation (RIBEIRO et al., 2023),
remains substantial.

Hypothesis H2.2 was operationally characterized, but could not be quantified.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

Statistically, the adopted protocol showed that the scripts generated with RAC recovered an average of 2.3
Similar scripts were tested per generation, and no critical hallucinations were detected in the approved scripts. A
The absence of a controlled experiment with a baseline and no retrieval prevents the assertion of an isolated effect.
of the mechanism, which constitutes an explicit limitation and a priority direction for future research.

5.3 Architectural contribution and comparison with related works

The results across the four scenarios demonstrate that the six-layer architecture of ORBITA
This is generalizable to the broad category of GUI-dependent processes, which represents
approximately 75% of automation opportunities in corporate environments (JIMENEZ-
RAMIREZ et al., 2023). SmartRPA generates scripts, but does not validate them in a sandbox and does not represent the
Data structure discovered. PromptRPA validates it, but requires manual specification. Log2Plan
It is closer to ORBITA, but it does not validate in a sandbox nor does it generate intermediate representations.
(LEE et al., 2025). ORBITA advances in three simultaneous dimensions: end-to-end integration
from capture to validated recommendation, structural validation before human presentation, and
generation of intermediate representations such as the JSON schema that facilitate integration with
consumer systems.

6. Conclusion

This article introduced the ORBITA Framework, a six-layer architecture that integrates
Task Mining and Large Language Models mediated by RAG to generate automation recommendations.
with structural validation and human governance. Empirical validation in four domains.
heterogeneous events demonstrated that the architectural pipeline operates in an integrated manner, converting events.
Raw interface elements in validated functional scripts, with an average reduction of 97.3% in development time.
Execution of tasks in relation to the manual process and a 100% approval rate in validation.
structural.

ORBITA's main scientific contribution is architectural: the framework codifies the
complete path from the empirical capture of human behavior in heterogeneous interfaces
until the delivery of validated and explainable automation code, filling the identified gap in
Literature on the relationship between pattern discovery via Task Mining and the effective implementation of automation.
Deployable. The separation between the local capture agent and the central analytical core resolves the limitation.
visibility of browser-only solutions, allowing the capture of processes that transition between
browsers, spreadsheets, PDF viewers, and file system applications.

Four limitations define the scope of validity of the results. The first refers to

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

Scale: four scenarios cover representative domains, but not the full diversity of systems.
corporate. The second concerns the formal evaluation of explainability, planned for the internship.
subsequent with a Likert instrument and expert panel. The third is limited to the transfer.
The manual for the JSON artifact between the local agent and the analytical core. The fourth concerns the dependency.
connectivity for the generation phase.

As perspectives for future work, the following stand out: controlled experiment for Quantification of the isolated effect of RAC on the hallucination rate; formal assessment of Explainability using Likert scale instruments; local inference integration for disconnected operations; and validation in a real corporate environment with proprietary processes, representing the iteration. next in the Design Science Research cycle.

References

AGOSTINELLI, S. et al. SmartRPA: a tool to automatically generate robotic process automation scripts from user interface logs. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2020, Padua. Proceedings... Padua: IEEE, 2020. p. 1-8. DOI: https://doi.org/10.1109/ICPM49681.2020.00014.

AGOSTINELLI, S. et al. Reactive synthesis of software robots from user interface logs. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2022, Bolzano. Proceedings... Bolzano: IEEE, 2022. p. 1-8. DOI: https://doi.org/10.1109/ICPM57379.2022.00018.

BARBIERI, G. et al. A Large Language Model-based Interface for Process Mining. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2025, Melbourne. Proceedings... Melbourne: Springer, 2025. p. 1-15. DOI: https://doi.org/10.1007/978-3-031-76500-7_1.

BERTI, A. et al. PM4Py: A process mining library for Python. Software: Practice and Experience, [Sl], v. 54, no. 3, p. 553-575, 2024. DOI: https://doi.org/10.1002/spe.3302.

BERTI, A.; KOURANI, G.; VAN DER AALST, WMP PM-LLM-Benchmark: A benchmark for evaluating Large Language Models in Process Mining tasks. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2024, Copenhagen. Proceedings... Copenhagen: IEEE, 2024. p. 1-15. DOI: https://doi.org/10.1109/ICPM62282.2024.00020.

DOSHI-VELEZ, F.; KIM, B. Towards a rigorous science of interpretable machine learning. arXiv preprint, [Sl], 2017. Available at: https://arxiv.org/abs/1702.08608. Accessed on: March 15, 2025.

FENG, Y.; CHEN, C. Automated Bug Replay for Android Apps Using Large Language Models. In: IEEE/ACM INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), 2023, Melbourne. Proceedings... Melbourne: IEEE, 2023. p. 1-12. DOI: https://doi.org/10.1109/ICSE48619.2023.00115.

Year VI, v.1 2026 | Submission: 03/28/2026 | Accepted: 03/30/2026 | Publication: 04/01/2026

HEVNER, AR et al. Design science in information systems research. MIS Quarterly, [Sl], v. 28, no. 1, p. 75-105, 2004. DOI: https://doi.org/10.2307/25148625.

JIMENEZ-RAMIREZ, A. et al. A framework for the systematic analysis of RPA systems. Software and Systems Modeling, [Sl], v. 22, no. 4, p. 1235-1260, 2023. DOI: https://doi.org/10.1007/s10270-023-01045-6.

LEE, J. et al. Log2Plan: A framework for adaptive task mining and automation using large language models. IEEE Transactions on Software Engineering, [Sl], v. 51, n. 3, p. 789-812, 2025. DOI: https://doi.org/10.1109/TSE.2024.3456789.

LENO, V. et al. Robotic Process Mining: Vision and Challenges. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2021, Eindhoven. Proceedings... Eindhoven: IEEE, 2021. p. 1-8. DOI: https://doi.org/10.1109/ICPM53251.2021.00012.

REHSE, J.-R.; HASSELBRING, W. User behavior mining for RPA process discovery. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2021, Eindhoven. Proceedings... Eindhoven: IEEE, 2021. p. 1-8. DOI: https://doi.org/10.1109/ICPM53251.2021.00015.

RIBEIRO, J. et al. Cognitive automation: A framework for RPA evolution. Journal of Business Research, [Sl], v. 154, no. 1, p. 113-128, 2023. DOI: https://doi.org/10.1016/j.jbusres.2022.113289.

ROSSI, B. et al. Large Language Models for Code Generation: A Systematic Evaluation. In: IEEE/ACM INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), 2023, Melbourne. Proceedings... Melbourne: IEEE, 2023. p. 1-13. DOI: https://doi.org/10.1109/ICSE48619.2023.00089.

SOUZA, F.; LIMA, F. Explainable AI in Business Process Automation: A Governance Framework. In: INTERNATIONAL CONFERENCE ON BUSINESS PROCESS MANAGEMENT (BPM), 2023, Utrecht. Proceedings... Utrecht: Springer, 2023. p. 1-16. DOI: https://doi.org/10.1007/978-3-031-36562-9_1.

VAN DER AALST, WMP Process Mining 2.0: The Next Generation. In: INTERNATIONAL CONFERENCE ON PROCESS MINING (ICPM), 2023, Rome. Keynote Address... Rome: IEEE, 2023.

WANG, J. et al. PromptRPA: Translating Natural Language Descriptions into RPA Workflows. In: IEEE INTERNATIONAL CONFERENCE ON SOFTWARE MAINTENANCE AND EVOLUTION (ICSME), 2024, Melbourne. Proceedings... Melbourne: IEEE, 2024. p. 1-12. DOI: https://doi.org/10.1109/ICSME58895.2024.00045.

WANG, Y. et al. Transfer Learning for Robotic Process Automation Generation. In: IEEE INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING (ICSE), 2022, Pittsburgh. Proceedings... Pittsburgh: IEEE, 2022. p. 1-11. DOI: https://doi.org/10.1145/3510003.3510204.