An Automated and Intelligent Interface Embracing Process Awareness into User Workspace

Minh Khoi Nguyen[1][a], Hanh Nhi Tran[1,2][b], Ileana Ober[1][c] and Razan Abualsaud[1][d]

[1]*Institut de Recherche en Informatique de Toulouse (IRIT), Toulouse, France*

[2]*Military Academy of Saint-Cyr Coëtquidan (AMSCC), CReC Saint-Cyr, France*

{*minh-khoi.nguyen, hanh-nhi.tran, ileana.ober, razan.abualsaud*}*@irit.fr*

Keywords: Artificial Intelligence, Process Management System, Process Monitoring, Collaborative Process.

Abstract: This paper presents an AI-augmented framework for automated and intelligent process monitoring, addressing the inefficiencies of manual progress reporting in Process Management Systems (PMS), which leads to potential inaccuracies and consumes valuable user time. Our research proposes a novel solution that bridges users' workspaces and PMS, enabling automatic progress reporting based on users' actions within their preferred tools. The core innovation of our framework *pMage* lies in employing Artificial Intelligence (AI) techniques to analyze and interpret sequences of user actions, translating them into accurate task progress updates, which significantly reduce manual input and enhance the accuracy of the reporting, thus making the integration of a PMS smoother and more effective. We demonstrate our framework's applicability through a case study that uses *pMage* to monitor a brake system manufacturing process with our prototype. As a smart interface, *pMage* provides a no-code solution to connect a wide range of user applications to various PMS via their respective APIs. This versatility ensures broad applicability across different organizational contexts and toolsets. Our AI-augmented framework offers a more reliable, efficient, and user-friendly approach than existing monitoring methods.

1 INTRODUCTION

In contemporary organizational environments, Process Management Systems (PMS) play a crucial role in monitoring and managing the progress of various tasks and projects. Traditionally, PMSs rely heavily on user inputs to track work progress, which presents significant challenges. Typically, users perform tasks using their preferred tools within their workspace and subsequently report their progress to the PMS manually. This method is fraught with issues, including the risk of inaccurate reporting and the considerable time investment required from users to log their activities.

Our research addresses this critical problem by proposing an innovative intermediate framework that acts as an intelligent interface between PMS and process participants' diverse workspaces. The framework's primary objective is to enable automatic and real-time process progress reporting by capturing and interpreting users' actions within their preferred tools. This seamless integration eliminates the need for manual updates, enhances tracking accuracy, and provides real-time visibility into process execution. By bridging the gap between user activities and process management, our framework aims to significantly improve the efficiency and timeliness of process monitoring, ultimately leading to more effective process management and decision-making.

The rest of this paper is organized as follows: Section 2 introduces the research questions and the contribution of this paper. The related works and current approaches are introduced in Section 3. Section 4 provides an overview of the core of our solution *pMage*. Section 5 focuses on our AI-driven solutions for offering a smart interface enhancement *pMage* that supports automated and intelligent process monitoring. Section 6 evaluates the proposed framework in the context of the case study on the process of manufacturing a brake system for a self-driving car, while Section 7 discusses the strengths and weaknesses of our approach and suggests directions for future research.

[a] https://orcid.org/0000-0001-5412-7259
[b] https://orcid.org/0000-0003-0868-1253
[c] https://orcid.org/0000-0001-9338-8187
[d] https://orcid.org/0009-0007-4895-1242

Nguyen, M. K., Tran, H. N., Ober, I. and Abualsaud, R.
An Automated and Intelligent Interface Embracing Process Awareness into User Workspace.
DOI: 10.5220/0013370200003896

In *Proceedings of the 13th International Conference on Model-Based Software and Systems Engineering (MODELSWARD 2025)*, pages 97-108
ISBN: 978-989-758-729-0; ISSN: 2184-4348

2 RESEARCH QUESTIONS

Defining a framework that allows to seamlessly integrate user workspace and process execution raises the following research questions:

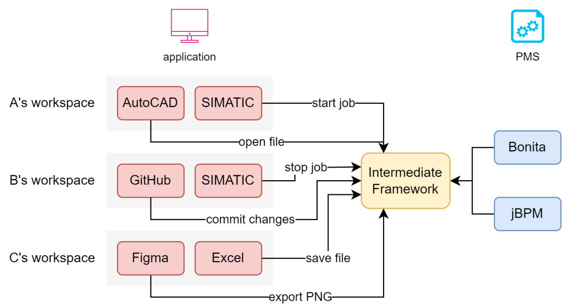

Figure 1: The intermediate framework to integrate end-user workspaces evoking diverse actions with PMSs.

RQ1. How to **manage and exploit the heterogeneity of tools and PMS** ? As shown in Fig. 1, the framework must be versatile enough to interface with various software environments (e.g. AutoCAD, GitHub, SIMATIC, etc.), each with its own data structures, APIs, and communication protocols. Balancing the need for a standardized approach with the flexibility to accommodate this diversity of tools is a complex undertaking.

RQ2. How to accurately **extract high-level task completion from the numerous low-level actions** users perform within their workspace. For example in Fig. 1, in a particular process execution, which task corresponds to the action of `start job` in SIMATIC? This requires sophisticated interpretation of user behaviors across various applications having their unique interface and functionality.

In our previous work (Nguyen et al., 2024), we developed a metamodel that abstracts tool families, enabling a homogeneous connection mechanism via tool APIs to manage tool heterogeneity. Inspired by the vision of (Dumas et al., 2023) to integrate more AI techniques into a business PMS to activate the intelligent adaptability and self-improvement of a business process, we also implemented an ontology-based mechanism to map low-level application actions to high-level tasks to solve the second research question. This ontology was built semi-automatically by extracting information from past projects (process names, task names, artifacts, actor roles, etc.) using an AI-augmented process mining framework (Nguyen et al., 2023). However, the extracted information from historical project data is often limited and potentially biased, providing inadequate contextual information on user actions, and leading to inaccuracies in mapping these actions to high-level tasks.

This paper presents two major contributions to automated process monitoring, providing machine intelligence to improve human productivity (Van der Aalst, 2021).

C1. We introduce an innovative procedure for **discovering appropriate connection configurations among heterogeneous tools by leveraging historical connection data**. This contribution enhances the framework's ability to adapt to diverse toolsets and organizational contexts, further streamlining the integration process and improving the overall flexibility of our framework.

C2. We **replace the previous ontology-based solution with a more sophisticated and powerful AI model** to infer task progression from user actions with higher precision. This AI model is trained using enriched data from past projects and web sources, significantly expanding the knowledge base and reducing potential biases.

These advancements represent a substantial leap forward in the accuracy, adaptability, and efficiency of automated process monitoring.

3 RELATED WORKS

Various approaches aim to provide the user with a smooth embrace of process management. While (Dueñas et al., 2018) proposed gathering information on the work progress of the project, (Delgado et al., 2016) introduced a central generic portal to engage different PMSs. Furthermore, some approaches propose a platform that allows monitoring the user work progress and updating the state of the process instance simultaneously (Baresi et al., 2017; Baresi et al., 2016).

To ensure users can continuously and transparently utilize a familiar PMS, (Delgado et al., 2016) proposed a generic PMS user portal designed to integrate with various concrete PMSs. This portal is built on a unified data model and a generic process engine API, enabling it to offer functions tailored to specific business contexts while minimizing disruptions caused by changes in the underlying PMS. The portal consists of two layers: a *presentation layer*, which defines the generic user interface, and an *access layer*, which facilitates the connection between the portal and specific process engines (e.g., Bonita, Activiti, and Bizagi). Although the proposed portal provides comprehensive PMS functionality and reduces the time required for training and usage, it does not fully free users from task reporting, limiting their ability to focus entirely on their primary working applications.

The need for an efficient mechanism to align specific work with the process model in a PMS has spurred various research efforts. (Cohn and Hull, 2009) highlights that the state of a process instance is inherently tied to the state of the business entities (or artifacts) involved and produced during its execution. Each artifact is characterized by an information schema and a life cycle, which defines how it evolves throughout the process (Dumas, 2011). Building on this concept, (Baresi et al., 2017) introduces an artifact-driven process monitoring platform called *mArtifact*, capable of flagging affected activities during violations without interrupting the monitoring process. This platform proves particularly effective for collaborative, multi-site environments. Similarly, (Baresi et al., 2016) proposes leveraging Smart Objects to monitor object states in cross-organization business processes. This approach, grounded in the GSM (Guard-Stage-Milestone) framework, enables the comparison of process instances against their models to detect control-flow violations and activity faults. To address storage limitations, Smart Objects only retain information about the activities that users intend to monitor.

To facilitate the management of the software development process, (Dueñas et al., 2018) introduced Perceval, a platform designed to automatically and incrementally collect data from various tools related to open-source development and present it through a centralized dashboard. Perceval is intended to be highly extensible, enabling cross-cutting analysis and providing incremental updates—particularly valuable for analyzing large software projects. However, Perceval does not handle data storage or analysis itself, so these tasks are delegated to other tools. Moreover, it lacks a concrete method for integration with specific PMSs, offering no direct support for end-users in task reporting.

These approaches rely on built-in platforms to manage processes, which require significant additional effort to customize for specific needs, domains, and contexts. While they offer the advantage of being adaptable to a user's unique requirements, they incur substantial costs in terms of the time and effort needed for development, testing, and deployment.

On the other hand, limited research has focused on leveraging machine learning or AI to enhance process awareness within user workspaces. (Weinzierl et al., 2024) highlights numerous studies that integrate and innovate machine learning and AI techniques across various aspects of business process management, including process identification, process discovery, process analysis, process redesign, process implementation, and process monitoring. Among these, process monitoring - most closely related to process awareness - commonly assumes the availability of perfectly structured event log data. Research in process monitoring has primarily explored methods for extracting features from online event logs (Leontjeva et al., 2015), dynamically reconstructing process models, and predicting process execution outcomes (Evermann et al., 2017; Metzger et al., 2019). In contrast, our proposed framework tackles the critical challenge of directly capturing event logs that reflect user behavior during the process, specifically within the user workspace.

4 pMage OVERVIEW

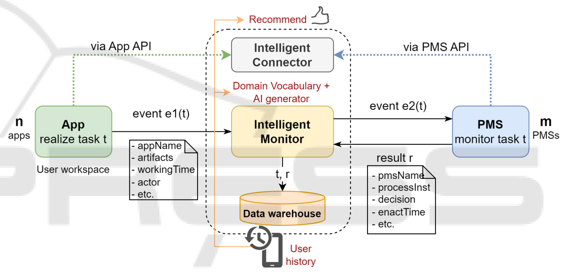

Figure 2: Architecture of pMage.

The architecture of pMage (illustrated in Fig. 2) has three main components: (1) an *Intelligent Connector* – that archives integration configurations and establishes connections between end-user applications and PMSs, (2) an *Intelligent Monitor* – that reports the progress of the impacted high-level tasks in the PMS from low-level user actions in their applications, and (3) a *Data Warehouse* – that keeps track of task executions for further analysis and learning.

The core component of our framework is the *Intelligent Monitor*, which acts as a bridge between user applications and the PMS to interpret low-level user actions to high-level task progress within the PMS. It is responsible for the following process ensuring that each user action is accurately reflected in the PMS in real-time without requiring manual updates:

- *Event Triggering*: as shown in Fig. 3, when a user performs an action in their application, the application triggers an event *e1* within the user's workspace.
- *Action Detection*: The *Intelligent Monitor* detects this triggered event *e1*, signaling that a user action

has occurred.

- *Action Interpretation*: To interpret this action, the *Intelligent Monitor* consults the Action Linkage table (c.f. Fig. 3). This table is a crucial component of our framework, storing precise mappings between low-level user actions in the workspace and corresponding high-level task actions in the PMS (mapping from App Event *e1* to PMS Event *e2* on task *t*).
- *Task Identification*: By referencing the Action Linkage table, the *Intelligent Monitor* identifies the specific task *t* associated with the user's action.
- *Progress Determination*: Based on the mapping, the *Intelligent Monitor* determines the progress made on the identified task *t* and then triggers the corresponding PMS event *e2* to update the task's progress accordingly.

App Event e1	Context Info	PMS Event e2	Task t
INFO tag start	brake_system_design	startTask	Design brake system blueprint
INFO tag complete	brake_system_design	endTask	Design brake system blueprint
INFO tag start	brake_activation	startTask	Implement Remote Sensing Devices braking activation

Figure 3: Action Linkage table provides the mapping references enabling identifying the impacted high-level task from the low-level user action.

The necessity for an accurate and comprehensive Action Linkage table comes from its role of enabling our framework to automatically interpret user actions and translate them into meaningful task progress updates. Yet, the creation of the Action Linkage table presents significant challenges and opportunities for innovation.

The Action Linkage table handles four aspects (columns in Fig. 3), each playing a specific role in the action-to-task mapping process. First, the event *e1*, triggered by a low-level action performed by the user within their workspace. It serves as the starting point for the mapping process, capturing the initial user interaction that needs to be interpreted. Second, the *contextual information* of the user action is essential for identifying the related task *t* in the PMS affected by the user's action. Furthermore, the event *e2* triggers an action in the PMS to update the relevant task *t*.

Traditionally, filling in such a table would require users to manually define the mappings based on their understanding of the tools and tasks involved in the process. Although this method can be precise, it is time-consuming and prone to human error, particularly when managing numerous actions across multiple tools. Additionally, as processes evolve and new tools are introduced, manually updating the table becomes increasingly difficult. This approach also demands extensive expertise in tools and processes concurrently, which may not always be readily accessible.

To address these challenges and facilitate the use of our framework, our objective is to generate the Action Linkage table to automate the mapping process, thereby reducing the time and effort required from users. While the data for the application event *e1* and PMS event *e2* can be extracted from the configurations of the user tool and the PMS respectively, filling the second and the fourth columns is more complicated. The second column containing contextual information about the user action is crucial for accurate task identification for filling the fourth column. This context is vital because the same user action can be associated with different tasks depending on the circumstances in which it is performed. For instance, saving a file might update a *document creation* task in one context, but signal the completion of a *review process* in another.

Determining the contextual information and corresponding process tasks requires an expert with deep knowledge of both the tool and the process. To overcome this dependency and enhance efficiency, we have developed an AI model that learns to provide this contextual information automatically (see Section 5). This AI-driven approach aims to replicate the nuanced understanding of an expert, enabling the framework to accurately discern the appropriate context for each user action and associate it with the correct task. By structuring the Action Linkage table in this manner and leveraging AI for context determination, our framework efficiently and accurately maps user actions to task updates. This approach significantly reduces the need for manual configuration while maintaining the necessary context awareness for precise task mapping, even when identical user actions correspond to different tasks based on their context.

5 DEVELOPMENT OF INTELLIGENT CONNECTOR AND INTELLIGENT MONITOR

This section details the two main advancements of our framework, which address the research questions

RQ1 and *RQ2*. First, we present the *Intelligent Connector*, which enables more efficient connection establishment (Sec. 5.1). Subsequently, we describe the *Intelligent Monitor*, which enables automated action interpretation and intelligent task monitoring (Sec. 5.2).

5.1 Intelligent Connector

Integrating applications and PMSs to monitor process execution within a project requires users to input essential information, such as the *application's name*, *project directory location*, and *login username*. This task becomes tedious and time-consuming when repeated for multiple projects.

To streamline this process, we leverage past connection data from the *Data Warehouse* to simplify new connection setups. This approach particularly benefits new users by suggesting common applications and PMS configurations for specific domains. These suggestions help users identify appropriate tools and provide necessary configuration and event profiles for intelligent monitoring.

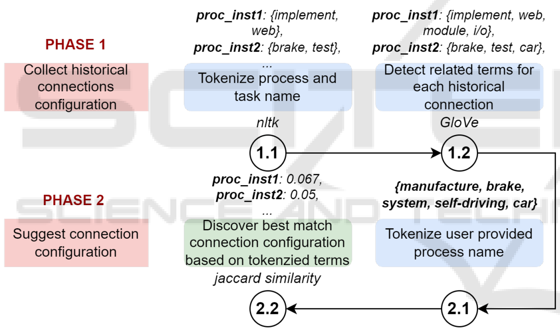

Figure 4: Workflow of *Intelligent Connector*.

The detailed method for implementing this approach is illustrated in Fig. 4 and Algo. 1. The main challenge is identifying the key information needed for effective configuration suggestions. Users expect an automatic field population based on the project name. Thus, the core of our suggestion process relies on **keyword correlation** between the new project name and historical projects' representative terms – project, task, and artifact names.

Our approach involves two key phases:

P1. Collecting Historical Connections Configuration

S1.1. By applying Natural Language Processing (NLP) including tokenizing, removing punctuation, stopwords, and lemmatizing, we collect representative terms of past connections from their process, task, and artifact names.

Algorithm 1: Connection Configuration Suggestion.

```
Input: username, processDesc, pastConnections
Output: bestConfig
forall connection ∈ pastConnections do
    representTerms :=
      nlp(connection.processName ∪
      connection.taskNames ∪
      connection.artifacts)
    forall term ∈ representTerms do
        relatedWords := findRelatedWords(term)
        extendedTerms.add(relatedWords)
    end
    connection.representTerms := extendedTerms
end
candidates := ∅
forall connection ∈ pastConnections do
    matchingWords :=
      findMatchingWords(processDesc,
      connection.representTerms)
    candidates.put(connection, matchingWords)
end
bestConfig := findUserPastConnection(username,
  candidates)
if bestConfig is null then
    bestConfig := findBestCandidate(candidates)
end
```

S1.2. These terms are expanded with their close meanings and synonyms to increase the potential for optimal configuration matches in P2.

P2. Suggesting Connection Configuration

S2.1. For each connection request, we extract terms from the new project name using similar NLP techniques.

S2.2. Compute Jaccard similarity between the new project and historical projects. Configurations created by the same user are prioritized for recommendations. Otherwise, the historical project with the highest Jaccard similarity becomes the suggested configuration for the new project.

$$JaccardSimilarity(A,B) = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

with A and B are the set of terms from the new project and the historical connection respectively.

5.2 Intelligent Monitoring

As explained in Section 4, the primary challenge for the *Intelligent Monitor* lies in generating the Action Linkage table, which captures the complex mappings between low-level user actions and high-level task updates, requiring a nuanced understanding of both the user's workspace and the process management context.

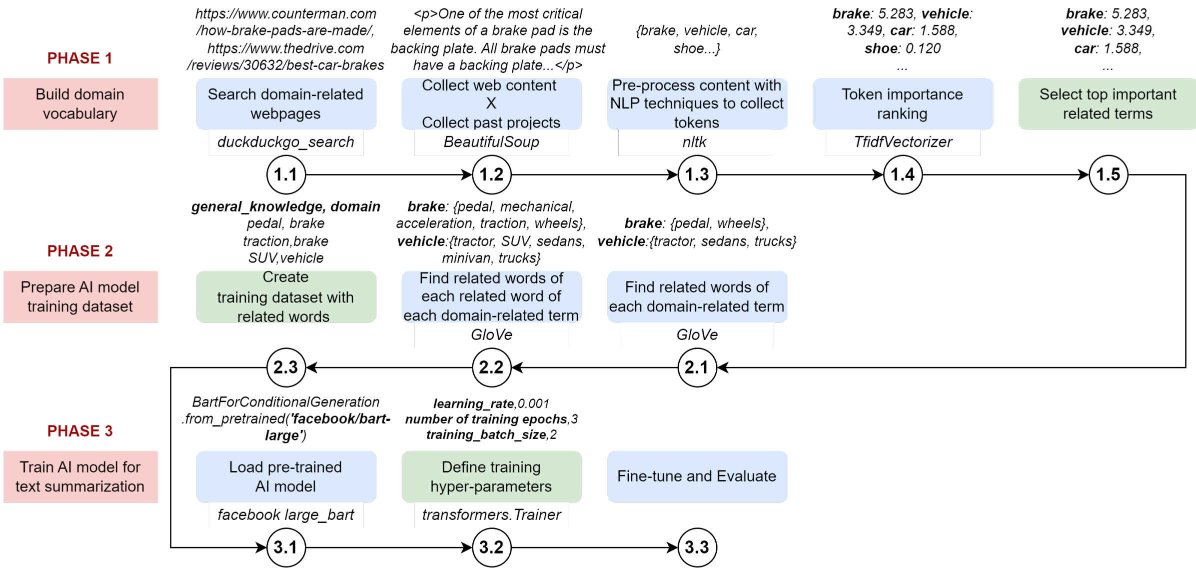

Figure 5: Workflow of fine-tuning AI model for *Intelligent Monitoring*.

To address the limitations of the previous solution (Nguyen et al., 2023), we leverage AI techniques more extensively to enhance the generation of the Action Linkage table. We propose using an AI model, which plays the role of a domain expert, to automatically generate the context information of the user's actions in the Action Linkage table. For this purpose, we enrich a pre-trained AI model's base knowledge with domain-specific vocabulary. While pre-trained AI models have learned a vast array of common words, enabling general information recognition, the addition of domain-related vocabulary provides a focused layer of understanding. This fine-tuned AI model unveils more appropriate contextual information, allowing for precise mapping and improved comprehension within the target domain. By learning from a broader, more diverse set of data and examples, the AI model generates more reliable and stable context information for the Action Linkage of each connection user application-PMS within the context of a given process.

However, implementing an AI-based solution for generating the Action Linkage table has its own challenges. Gathering a sufficient and diverse dataset of user actions and corresponding task progress for training the AI model is a significant hurdle. Developing algorithms that can understand the context of actions across different tools and processes adds another layer of complexity. Creating a model that can generalize well across various domains and types of processes is crucial for the framework's versatility. Finally, ensuring that the AI-generated mappings are accurate and reliable for real-world use is paramount.

Figure 5 presents the three key proposed phases for developing *Intelligent Monitoring*, along with their detailed steps and the illustrated results from the experiment described in Section 6:

P1. Build Domain Vocabulary: Learning domain vocabulary from both domain-related web pages and past projects has (1) greater accessibility to up-to-date terms, (2) a more diverse and less biased vocabulary, and (3) a balanced integration of user behaviors and domain knowledge. We employ the following process to construct the domain vocabulary:

S1.1. Search for domain-related web pages using keywords.

S1.2. Extract content from (1) the collected domain-related web pages, and (2) historical project connections.

S1.3. Pre-process the extracted terms using the same NLP techniques as for the workflow of *Intelligent Connector*.

S1.4. Compute and rank the importance of each term by calculating its TF-IDF value across all web page content. TF-IDF (Term Frequency Inverse Document Frequency) measures the relevance of a word within a corpus. The importance increases proportionally to the number of times a term t appears in a web page p ($N(t)_p$) relative to the total terms in the page (N_p), and is adjusted by the term frequency ($N(t)$) across the entire corpus of N

web pages.

$$
\begin{aligned}
TF-IDF(t) &= \sum_{p} TF-IDF(t,p) \\
&= \sum_{p} \frac{N(t)_p}{N_p} * \log(\frac{N}{N(t)})
\end{aligned} \tag{2}
$$

S1.5. Select a fixed number of terms with the highest importance values as the domain vocabulary.

P2. Prepare AI Model Training Dataset: This phase is crucial for guiding the AI model to generate contextual information for the Action Linkage table using domain-specific terms based on the task names.

S2.1. For each term in the domain vocabulary, we use GloVe (Global Vectors for Word Representation) (Pennington et al., 2014) to identify synonyms and related words.

S2.2. Extend the search to find related words at a second level (related words of the initial related words). The ideal approach would be an infinite loop of searches until no new words are found, however, we limit our search to the second level due to resource and time constraints. Duplicate terms are excluded, resulting in an enriched set of related words for each domain term.

S2.3. Construct a training dataset to fine-tune the pre-built AI model for text summarization. The AI model aims to generate domain-specific terms from the input process name.

P3. Train AI Model for Text Summarization: This final phase focuses on training the AI model to understand domain-specific terms and summarize input text into domain-related keywords, which will be used as contextual information in the Action Linkage table.

S3.1. Load a pre-built Transformer model trained on a large dataset of diverse tasks. This pre-trained model enables transfer learning, where knowledge from one task is adapted to a different but related task by fine-tuning the model on a smaller, task-specific dataset.

S3.2. Define hyper-parameters suitable for the pre-trained AI model and the specialized text summarization task (e.g. Table 2), which involves balancing our computing resources with recommendations from other studies on the AI model and the specialized task.

S3.3. Train the AI model with the defined hyper-parameters using the dataset created in P2: the general knowledge terms as the *input* and the domain-related terms as the targeted *output*. Each time the fine-tuned AI model meets a text, it recognizes the general knowledge terms and returns the domain-related terms to generate better contextual information for the Action Linkage table.

6 EVALUATION

The evaluation aims to validate the proposed intelligent components by assessing their applicability and efficiency in suggesting new integrating connections, detecting context information from user applications, and scalability to handle large datasets and complex process configurations when applying to case study *Manufacture Brake System for a Self-driving Car*.

6.1 Evaluation Method

The evaluation is carried out as follows:
Case Study: we imitate the procedure of integrating the process execution of *Manufacture Brake System for a Self-driving Car* (c.f. Fig. 6) into our proposed framework, including the step of connection establishment and setting up the Action Linkage table for monitoring user action.

This process represents a research and development workflow, customized for the manufacturing of a self-driving car model. Regarding the process tasks, the tasks *Design brake system blueprint*, *Implement RSDs braking activation*, *Connect RSD with brake control system*, and *Assemble the system into a prototype vehicle* are performed once for each car model individually. In contrast, the task *Assemble the system into the final version* is executed multiple times on an assembly line that handles multiple vehicles of the same model. In terms of roles, the process involves two key contributors: the *Mechanical Engineer* and the *Embedded System Developer*. A notable characteristic of this case study is that the process is executed by distinct actors, each utilizing entirely different sets of applications and project management systems (PMS):

- The Embedded System Developer uses **Github** - a developer platform that allows creating, storing, and sharing code - for *Implementing remote sensing devices braking activation*
- The Mechanical Engineer uses **AutoCAD** - a general drafting and design application to prepare technical drawings - for *designing brake system blueprint*

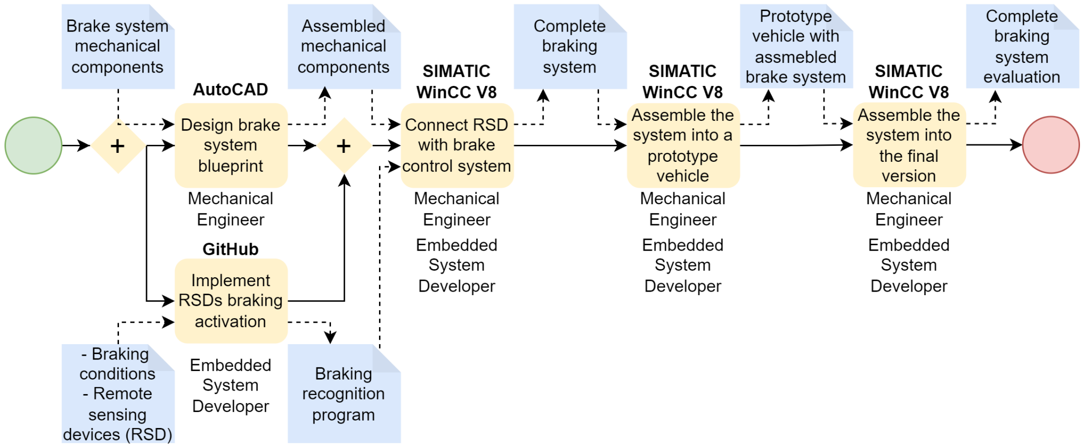

Figure 6: Process model of Manufacturing Brake System for a Self-driving car.

- **SIMATIC WinCC V8** - a process visualization system enabling seamless monitoring and operation of automated processes - is used by both the Embedded System Developer and the Mechanical Engineer for assembly tasks.

The case study highlights the challenges of managing complex communication between participating applications and PMSs. A connection is needed for each pair of user applications and PMS. Additionally, a mapping from low-level user actions to high-level task updates is required for each connection. The presence of numerous connections within a single project increases both the technical complexity of establishing these connections and the potential ambiguity in action interpretation.

Simulation: Synthetic event logs are generated to simulate various process definitions to examine the proposed *Intelligent Connector* of the framework.

For evaluating the proposed framework performance, we use 3 different metrics:

- **Jaccard Similarity:** to assess the similarity between the new project name and the suggested historical project configuration of the *Intelligent Connector*.
- **ROUGE** (Recall-Oriented Understudy for Gisting Evaluation): a set of metrics used for evaluating the summaries' quality by measuring the overlap between the generated summary and the original text of the Action Linkage table. We focus on ROUGE-1, and ROUGE-L:
  - **ROUGE-1:** compares the number of matching individual words between the summary and the original text.
  - **ROUGE-L:** measures the longest common subsequence (LCS) between the generated summary and the original text but does not require consecutive words, making it a more flexible and lenient similarity measure.
- **BERT Score:** uses contextual embeddings from pre-trained Transformer-based models to compute the similarity between the generated summary and the original text of the Action Linkage table.

We expect our framework to achieve a high similarity score in the connection suggestion and the generated contextual information. The similarity scores reflect the effectiveness of a solution and help discover the impact of preparing the training dataset for the AI model, and the diversity of history connections on the framework performance.

6.2 Data Sources

We have made available datasets necessary for developing the intelligent components of pMage concentrated on the case study.

6.2.1 Historical Connections

For the time we run our case study, this data, theoretically collected by *pMage*, is not enough to construct a comprehensive dataset that spans various projects across multiple domains. Consequently, simulating user connections is a viable alternative to real data, forming the basis for suggesting configurations.

As outlined in Algorithm 1, the process and its task descriptions are key elements in identifying the appropriate configuration. We design the simulated data with these considerations:

Seven different process definitions were simulated: *Modify Test-bench Wiring*, *Implement E-commerce Web Application*, *Manufacture Brake System for a Self-driving Car*, *Certify Car Brake*, *Resignation Procedure*, *Item Ordering Process*, and *Apply*

Table 1: Key aspects of the simulated historical connections.

Number of process definitions	7
Number of users	12
Number of process instances	1338
Number of applications	6
Number of connections	2000
Number of PMSs	3

for Schengen Visa. Each process involves up to two users for its execution. Additionally, six applications were designed and tailored for a specific process or usable across various processes.

6.2.2 Domain Vocabulary and Training Dataset for AI Model

We use the DuckDuckGo search engine API (`duckduckgo_search`) to find domain-related web pages using the keywords *manufacture car brake*. DuckDuckGo provides consistent results for all users, avoiding personalized results based on search history and ensuring stable and unbiased search outcomes. We use `BeautifulSoup` to scrape content from the top 50 related web pages. We also gather project, task, and artifact names from historical projects. Using `nltk` (natural language toolkit), we preprocess all collected texts by removing punctuation and stopwords and performing lemmatization. Finally, we rank the preprocessed texts using `TfidfVectorizer` from `sklearn`, creating a dataset of 500 domain-related terms for AI model training.

We discover each domain-related term its related words using GloVe. To enhance the AI model's capability to recognize domain-related terms amidst common terms, we include not only the related words of the domain-related words but also the related words of those related words. This approach results in a rich and diverse set of domain-specific terms.

The pre-trained AI model used in our experiment is Large BART from https://huggingface.co/facebook/bart-large (Lewis et al., 2019). BART uses a *transformer-based encoder-decoder* architecture with a bidirectional encoder (like BERT (Devlin et al., 2019)) and a left-to-right decoder (like GPT). The pretraining task involves randomly shuffling the order of the original sentences and a novel in-filling scheme, where spans of text are replaced with a single mask token. The hyper-parameters used for fine-tuning BART are detailed in Table 2.

Table 2: Hyper-parameters for fine-tuning Large BART.

Learning rate	0.001
Number of training epochs	3
Training batch size	2
Evaluation batch size	2
Loss function	Cross-Entropy
Weight decay	0.01

6.3 Project Specification

As described in Fig. 4, the results of the connection configuration recommendation rely on the terms in the user's process description. Our approach also prioritizes suggesting the user's historical connection rather than from the other users. For this reason, to examine the performance of our approach in suggesting connection configuration, we test different setups:

- Project name:
  - contains terms only included in the historical connections: *Manufacture Brake system for Self-driving Car*
  - contains some terms included in the historical connections: *Brake system for new electric Peugeot*.
  - does not contain any terms included in any historical connections: *Develop Anti-cheat for FPS games*
- User with (1) or without (2) historical connections

6.4 Results and Discussions

6.4.1 Connection Configuration Suggestion

Deduced from Table 3, we can suggest a connection configuration for projects whose names are composed entirely of terms archived in historical connections. Although the Jaccard similarity score is relatively low (0.067), indicating a small fraction of shared terms compared to the total terms, this is acceptable as we have enriched the representative terms for each historical connection (see Fig. 4). Conversely, projects with partially matching names to historical connections have a lower Jaccard similarity score of 0.025, particularly for a different project type like *Certifying Car Brake*. Terms such as *Peugeot* and *electric* contribute to the lower similarity score since they are not present in the representative terms, meanwhile the remaining terms are more akin to the *Certify Car Brake* project than to a *Manufacture Brake System for Self-driving Car* project. Projects containing entirely different terms from the historical connections result in a perfect 0 similarity score, making it impossible to suggest a connection configuration if there are no similarities between the project's name and the historical connections.

The results are consistent for existing users with historical connections and new users. The main distinction for existing users is that pMage can suggest authorization information, such as login credentials, passwords, or personal tokens. pMage assists all users, including those new to the framework, in adopting a process-aware working environment with greater ease and efficiency.

6.4.2 Action Linkage Generation

We selected the project name *Manufacture Brake System for Self-driving Car* with its suggested configuration to establish the connection between application *SIMATIC WinCC V8* and PMS *jBPM*. The next crucial step after establishing the connection is defining the Action Linkage table. Table 4 illustrates the generated Action Linkage table for this connection.

Table 3: Connection configuration suggestions on different setups.

Project name	**User**	**Configuration suggested**	**Jaccard score**
Manufacture Brake system for Self-driving Car	(1)	Manufacture self-driving Car Brake, app (SIMATIC WinCC V8, C:\Program Files(x86)\...), pms (jBPM, localhost/kie-server/...)	0.067
Brake system for new electric Peugeot	(1)	Certify Car Brake, app (SIMATIC WinCC V8, C:\Program Files(x86)\...), pms (jBPM, localhost/kie-server/...)	0.025
Develop Anti-cheat for FPS games	(1)	No configuration	0.000
Manufacture Brake system for Self-driving Car	(2)	Manufacture self-driving Car Brake, app (SIMATIC WinCC V8, C:\Program Files(x86)\...), pms (jBPM, localhost/kie-server/...)	0.067
Brake system for new electric Peugeot	(2)	Certify Car Brake, app (SIMATIC WinCC V8, C:\Program Files(x86)\...), pms (jBPM, localhost/kie-server/...)	0.025
Develop Anti-cheat for FPS games	(2)	No configuration	0.000

We defined the event profiles for each application specifying

1. the **method** of collecting the event,
2. the **location** of collecting the event, and
3. the **patterns** within the event where contextual information should appear

Below is an example of event profiles for *SIMATIC WinCC V8*, which includes two events: `INFO tag start` and `INFO tag complete`.

```
"event": [{"name": "INFO tag start", "method": "
    LOG",
        "apiInfo": "C:\\Program Files(x86)\\
            Siemens\\WinCC\\Diagnose",
        "important": "[{time}] INFO [{task}] {task
            } started for '{artifact}' by user '{
            userName}'."},
        {"name": "INFO tag complete", "method": "
            LOG",
        "apiInfo": "C:\\Program Files(x86)\\
            Siemens\\WinCC\\Diagnose",
        "important": "[{time}] INFO [{task}] {task
            } completed for '{artifact}' by user
            '{userName}'."}]
```

With event `INFO tag start`:

- The method for collecting the event is logging journal
- The location for collecting the event is C:\Program Files(x86)\Siemens\WinCC\Diagnose
- The patterns within the event where contextual information should appear is `[time] INFO [task] task started for 'artifact' by user 'userName'`. By analyzing these event patterns, we can collect contextual information (time, task name, artifact name, user name), which is important to create the Action Linkage table and coordinate the workflow of the target artifacts.

By analyzing historical connections, we identified common pairs of application and PMS events. For instance, in Table 4, the application event *INFO tag start* from SIMATIC WinCC V8 — indicating the initiation of a job identified by a specific tag — frequently corresponds with the PMS event *startTask* in jBPM, which starts the task in the PMS. Similarly, the relationship between *INFO tag complete* and *endTask* was also observed.

From Table 4, we can observe the importance of contextual information in determining the corresponding process task. The first and third lines of Table 4 have a similar pair of application and PMS event, `INFO tag start` and `startTask` respectively. The generated contextual information is designed to accurately reflect the meaning of each corresponding task while avoiding ambiguity between tasks with similar meanings. Our fine-tuned AI model can generate unique contextual information for each task.

Table 4: Action Linkage generation for the configuration of *Manufacture Brake system for Self-driving Car* project on *SIMATIC WinCC V8* application and *jBPM* PMS.

App Event	Context Info	PMS Event	Task
INFO tag start	brake system design	start-Task	Design brake system blueprint
INFO tag complete	brake system design	end-Task	Design brake system blueprint
INFO tag start	brake activation	start-Task	Implement Remote Sensing Devices braking activation
INFO tag complete	brake activation	end-Task	Implement Remote Sensing Devices braking activation
INFO tag start	brake Vehicle	start-Task	Connect Remote Sensing Devices with brake control system
INFO tag complete	brake Vehicle	end-Task	Connect Remote Sensing Devices with brake control system
INFO tag start	au-tom/brake	start-Task	Assemble the system into a prototype vehicle
INFO tag complete	au-tom/brake	end-Task	Assemble the system into a prototype vehicle
INFO tag start	technician Assemble	start-Task	Assemble the system into the final version
INFO tag complete	technician Assemble	end-Task	Assemble the system into the final version

Table 5: Similarity score for contextual information generated in the case study evaluation.

Contextual Info	ROUGE-1	ROUGE-L	BERT score
brake system design	0.857	0.571	0.600
brake activation	0.5	0.5	0.528
brake Vehicle	0.2	0.2	0.455
autom/brake	0.0	0.0	0.393
technician Assemble	0.222	0.222	0.462

For example, *brake_system_design* succinctly represents the task *Design brake system blueprint* with a ROUGE-1 similarity score of 0.857. The generated contextual information remains distinct even for tasks with similar meanings, such as *Assemble the system into a prototype vehicle* and *Assemble the system into the final version*. The generated Action Linkage table solves the *RQ2* by successfully mapping the low-level action in the applications and the high-level task in the PMS, allowing seamless process monitoring.

Furthermore, users can modify the Action Linkage table to suit their needs. For instance, if the roles of Mechanical Engineer and Embedded System Developer do not utilize SIMATIC WinCC V8 for the tasks *Design brake system blueprint* and *Implement Remote Sensing Devices braking activation* respectively, they can remove the related tuples from the generated Action Linkage table. Additionally, observed from Table 5, some generated contextual information values, such as *autom/brake* and *technician.Assemble*, have low similarity scores on the ROUGE-1 and ROUGE-L, but significantly better score on BERT score. The reason for such scores might come from the limited size of the original text. They might be unclear or hard to remember, potentially requiring human intervention to correct. The corrected terms become learning material for *pMage* to improve the *Intelligent Connector* suggestion and the *Intelligent Monitor* training data.

The case study illustrates the capabilities of our proposed framework, showcasing its practical benefits and efficiency in solving both research questions *RQ1* and *RQ2*. We are currently conducting additional process monitoring studies to evaluate the application of *pMage* in other domains.

7 CONCLUSION

In this study, we address the research questions of *(RQ1)* how to manage and exploit the heterogeneity of tools and PMS when establishing connections, and *(RQ2)* how to deduce high-level tasks from the numerous low-level actions. Through comprehensive analyses and the application of two advancements *Intelligent Connector* and *Intelligent Monitor*, our experiments' results demonstrate that the improved intermediate framework *pMage* not only leverages historical connections to suggest configurations for new projects using Jaccard similarity but also develops a more efficient mechanism for generating contextual information for the Action Linkage table by fine-tuning a pre-trained AI model using domain-related terms extracted from web pages and historical connections. Although the proposed solutions have some limitations including periodic rather than real-time domain vocabulary updates, as collecting and enriching the vocabulary and fine-tuning the AI model for specific tasks require considerable time, the findings suggest that intelligent components' integration is a viable and more effective solution for addressing the coordination between diverse tools and PMSs and embracing process awareness in user workspace, ultimately paving the way for further exploration and implementation in intelligent process management.

Our vision is to evolve towards a comprehensive large process model (Kampik et al., 2023). Future research contributions aim to use historical data for process mining and insight discovery, train AI models to recommend new process definitions without domain expertise and define an advanced framework with even more assistance for the setup phases for collaborative teams.

REFERENCES

Baresi, L., Di Ciccio, C., Mendling, J., Meroni, G., and Plebani, P. (2017). martifact: an artifact-driven process monitoring platform. In Robert Clarisó, Henrik Leopold, Jan Mendling, Wil M. P. van der Aalst, Akhil Kumar, Brian T. Pentland, Mathias Weske, editor, *Proceedings of the BPM Demo Track and BPM Dissertation Award co-located with 15th International Conference on Business Process Modeling (BPM 2017), September 13, 2017.*, pages 1 – 5, Barcelona, Spain. CEUR Workshop Proceedings.

Baresi, L., Meroni, G., and Plebani, P. (2016). A gsm-based approach for monitoring cross-organization business processes using smart objects. In *Business Process Management Workshops*, pages 389–400, Cham. Springer International Publishing.

Cohn, D. and Hull, R. (2009). Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.*, 32:3–9.

Delgado, A., Calegari, D., and Arrigoni, A. (2016). Towards a generic bpms user portal definition for the execution of business processes. *Electronic Notes in Theoretical Computer Science*, 329:39–59. CLEI 2016 - The Latin American Computing Conference.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dueñas, S., Cosentino, V., Robles, G., and Gonzalez-Barahona, J. M. (2018). Perceval: Software project data at your will. In *2018 IEEE/ACM 40th International Conference on Software Engineering: Companion (ICSE-Companion)*, pages 1–4.

Dumas, M. (2011). On the convergence of data and process engineering. In *Proceedings of the 15th International Conference on Advances in Databases and Information Systems*, ADBIS'11, page 19–26, Berlin, Heidelberg. Springer-Verlag.

Dumas, M., Fournier, F., Limonad, L., Marrella, A., Montali, M., Rehse, J.-R., Accorsi, R., Calvanese, D., De Giacomo, G., Fahland, D., Gal, A., La Rosa, M., Völzer, H., and Weber, I. (2023). AI-augmented business process management systems: A research manifesto. *ACM Trans. Manage. Inf. Syst.*, 14(1):11:1–11:19.

Evermann, J., Rehse, J.-R., and Fettke, P. (2017). Predicting process behaviour using deep learning. *Decision Support Systems*, 100:129–140. Smart Business Process Management.

Kampik, T., Warmuth, C., Rebmann, A., Agam, R., Egger, L. N. P., Gerber, A., Hoffart, J., Kolk, J., Herzig, P., Decker, G., van der Aa, H., Polyvyanyy, A., Rinderle-Ma, S., Weber, I., and Weidlich, M. (2023). Large process models: Business process management in the age of generative AI.

Leontjeva, A., Conforti, R., Di Francescomarino, C., Dumas, M., and Maggi, F. M. (2015). Complex symbolic sequence encodings for predictive monitoring of business processes. In Motahari-Nezhad, H. R., Recker, J., and Weidlich, M., editors, *Business Process Management*, pages 297–313, Cham. Springer International Publishing.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Metzger, A., Neubauer, A., Bohn, P., and Pohl, K. (2019). Proactive process adaptation using deep learning ensembles. In Giorgini, P. and Weber, B., editors, *Advanced Information Systems Engineering*, pages 547–562, Cham. Springer International Publishing.

Nguyen, M. K., Nhi Tran, H., Ober, I., and Abualsaud, R. (2024). Ai-augmented framework to enable process awareness in collaborative teams. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Yokohama, Tokyo. IEEE.

Nguyen, M. K., Tran, H. N., Abualsaud, R., and Ober, I. (2023). An AI-enhanced process mining framework for software process insights. In *2023 15th International Conference on Knowledge and Systems Engineering (KSE), October 18-20, 2023, Ha Noi, Vietnam*, pages 1–6, Hanoi, Vietnam. IEEE. ISSN: 2694-4804.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. volume 14, pages 1532–1543.

Van der Aalst, W. M. (2021). Hybrid intelligence: to automate or not to automate, that is the question. *International Journal of Information Systems and Project Management*, 9(2):5–20.

Weinzierl, S., Zilker, S., Dunzer, S., and Matzner, M. (2024). Machine learning in business process management: A systematic literature review. *Expert Systems with Applications*, 253:124181.